

Hierarchical Unambiguity*

Holger Spakowski[†]

Institut für Informatik
Heinrich-Heine-Universität Düsseldorf
40225 Düsseldorf, Germany
spakowsk@cs.uni-duesseldorf.de

Rahul Tripathi

Department of Computer Science and Engineering
University of South Florida
Tampa, FL 33620, USA
tripathi@cse.usf.edu

Abstract

We develop techniques to investigate relativized hierarchical unambiguous computation. We apply our techniques to generalize known constructs involving relativized unambiguity based complexity classes (UP and Promise-UP) to new constructs involving arbitrary higher levels of the relativized unambiguous polynomial hierarchy (UPH). Our techniques are developed on constraints imposed by hierarchical arrangement of *unambiguous* nondeterministic polynomial-time Turing machines, and so they differ substantially, in applicability and in nature, from standard methods (such as the switching lemma [Hås87]), which play roles in carrying out similar generalizations.

Aside from achieving these generalizations, we resolve a question posed by Cai, Hemachandra, and Vyskoč [CHV93] on an issue related to nonadaptive Turing access to UP and adaptive smart Turing access to Promise-UP.

1 Introduction

1.1 Background

Baker, Gill, and Solovay in their seminal paper [BGS75] introduced the concept of relativization in complexity theory, and showed that the bottom levels of the polynomial hierarchy P and NP separate in some relativized world. Baker and Selman [BS79] made progress in extending this relativized separation to the next levels of the polynomial

*A preliminary version of this paper was presented at the MFCS '06 conference.

[†]Supported in part by the DFG under grants RO 1202/9-1 and RO 1202/9-3.

hierarchy: They proved that there is a relativized world where $\Sigma_2^P \neq \Pi_2^P$. However, Baker and Selman [BS79] noted that their proof techniques do not apply at higher levels of the polynomial hierarchy because of certain constraints in their counting argument. Thus, it required the development of entirely different proof techniques for separating all the levels of the relativized polynomial hierarchy. The landmark paper by Furst, Saxe, and Sipser [FSS84] established the connection between the relativization of the polynomial hierarchy and lower bounds for small depth circuits computing certain functions. Techniques for proving such lower bounds were developed in a series of papers [FSS84, Sip83, Yao85, Hås87], which were motivated by questions about the relativized structure of the polynomial hierarchy. Yao [Yao85] finally succeeded in separating the levels of the relativized polynomial hierarchy by applying these new techniques. Håstad [Hås87] gave the most refined presentation of these techniques via the *switching lemma*. Even to date, Håstad's switching lemma [Hås87] is used as an essential tool to separate relativized hierarchies, composed of classes stacked one on top of another. (See, for instance, [Hås87, Ko89, BU98, ST] where the switching lemma is used as a strong tool for proving the feasibility of oracle constructions.)

A major contribution of our paper lies in demonstrating that known oracle constructions involving the initial levels of the unambiguous polynomial hierarchy (UPH) and the promise unambiguous polynomial hierarchy (UPH), i.e. UP and $P_s^{\text{Promise-UP}}$, respectively, can be extended to oracle constructions involving arbitrary higher levels of UPH by application only of pure counting arguments. In fact, it seems implausible to achieve these extensions by well-known techniques from circuit complexity (e.g., the switching lemma [Hås87] and the polynomial method surveyed in [Bei93, Reg97]).

The class UP is the unambiguous version of NP. UP has proved to be useful for instance in studying worst-case one-to-one one-way functions [Ko85, GS88], obtaining potential counterexamples to the Berman-Hartmanis isomorphism conjecture [JY85], and in the study of the complexity of closure properties of #P [OH93]. Lange and Rossmanith [LR94] generalized the notion of unambiguity to higher levels of the polynomial hierarchy. They introduced the following unambiguity based hierarchies: AUPH, UPH, and UPH . It is known that $AUPH \subseteq UPH \subseteq UPH \subseteq UAP$ [LR94, CGRS04], where UAP (unambiguous alternating polynomial-time) is the analog of UP for alternating polynomial-time Turing machines. These hierarchies received renewed interests in some recent papers (see, for instance, [ACRW04, CGRS04, ST, GT05]). Spakowski and Tripathi [ST], developing on circuit complexity-theoretic proof techniques of Sheu and Long [SL96], and of Ko [Ko89], obtained results on the relativized structure of these hierarchies. Spakowski and Tripathi [ST] proved that there is a relativized world where these hierarchies are infinite. They also proved that for each $k \geq 2$, there is a relativized world where these hierarchies collapse so that they have exactly k distinct levels and their k 'th levels collapse to PSPACE. The present paper supplements this investigation with a focus on the structure of the unambiguous polynomial hierarchy.

1.2 Results

We prove a combinatorial lemma (Lemma 3.1) and demonstrate its usefulness in generalizing known relativization results involving classes such as UP and Promise-UP to new relativization results that involve arbitrary levels of the unambiguous polynomial hierarchy (UPH).

In Subsection 4.1, we use Lemma 3.1 to construct relativized worlds in which certain inclusion relationships between bounded ambiguity classes ($UP_{O(1)}$ and FewP) and the levels of the unambiguous polynomial hierarchy (UPH) do not hold. Theorem 4.1 of this subsection subsumes an oracle result of Beigel [Bei89] for any constant $k \geq 1$ and Corollary 4.5 generalizes a result of Cai, Hemachandra, and Vyskoč [CHV93] from the case of $k = 2$ to the case of any arbitrary $k \geq 2$.

Subsection 4.2 studies the issue of simulating nonadaptive access to $U\Sigma_h^p$, the h 'th level of the unambiguous polynomial hierarchy, by adaptive access to $U\Sigma_h^p$. Theorem 4.7 of this subsection generalizes a result of Cai, Hemachandra, and Vyskoč [CHV92] from the case of $h = 1$ to the case of any arbitrary $h \geq 1$. Lemma 3.1 is used as a key tool for proving Theorem 4.7.

We improve upon Theorem 4.7 of Subsection 4.2 in Subsection 4.3. There are compelling reasons for the transition from Subsection 4.2 to Subsection 4.3, which we discuss in Subsection 4.3. Theorem 4.10 in that subsection not only resolves a question posed by Cai, Hemachandra, and Vyskoč [CHV93], but also generalizes one of their results. In particular, Theorem 4.10 holds for any total, polynomial-time computable and polynomially bounded function $k(\cdot)$ and arbitrary $h \geq 1$, while a similar result of Cai, Hemachandra, and Vyskoč [CHV93] holds only for any arbitrary *constant* k and $h = 1$. Lemma 3.1 is one of the ingredients in the proof of this theorem.

Subsection 4.4 investigates the complimentary issue of simulating adaptive access to $U\Sigma_h^p$ by nonadaptive access to $U\Sigma_h^p$. Theorem 4.13 of this subsection generalizes a result of Cai, Hemachandra, and Vyskoč [CHV93] from the case of $h = 1$ to the case of any arbitrary constant $h \geq 1$. Again, Lemma 3.1 is useful in making this generalization possible.

In Subsection 4.5, we study the notion of one-sided helping introduced by Ko [Ko87]. Corollary 4.17 of this subsection generalizes and improves one of the results of Cai, Hemachandra, and Vyskoč [CHV93].

Finally, in Section 5 we consider the possibility of imposing a more stringent restriction in the statement of Lemma 3.1. The investigation in this subsection leads to a generic oracle collapse of UPH to P under the assumption $P = NP$. This extends a result of Blum and Impagliazzo [BI87], which showed a generic oracle collapse of UP (the first level of UPH) to P assuming $P = NP$.

2 Preliminaries

2.1 Notations

Let \mathbb{N}^+ denote the set of positive integers. Σ denotes the alphabet $\{0, 1\}$. Let $[n] =_{df} \{1, 2, \dots, n\}$ for every $n \in \mathbb{N}^+$. NPTM stands for “nondeterministic polynomial-time Turing machine.” For every oracle NPTM N , oracle A , and string $x \in \Sigma^*$, we use the shorthand $N^A(x)$ for “the computation tree of N with oracle A on input x .” We fix a standard, polynomial-time computable and invertible, one-to-one, multiarity pairing function $\langle \cdot, \dots, \cdot \rangle$ throughout the paper. Let \circ denote the composition operator on functions. For any polynomial $p(\cdot)$ and integer $i \geq 1$, let $(p \circ)^i(\cdot)$ denote $\underbrace{p \circ p \circ \dots \circ p(\cdot)}_i$, i.e., the polynomial

obtained by i compositions of p . All polynomials $p(\cdot)$ appearing in this paper are without loss of generality nondecreasing and satisfy $p(n) \geq n$ for every $n \in \mathbb{N}^+$. Let σ be an equivalence relation on a set S . For each $x \in S$, the *equivalence class* $[x]$ of x determined by σ is $\{y \in S \mid x\sigma y\}$. The set S/σ of all equivalence classes determined by σ is called the *quotient set* determined by σ . For any set S , we use $\wp(S)$ to denote the *power set* of S , i.e., the set of all subsets of S . The *join* of two sets A and B over Σ is defined as $A \oplus B = \{0x \mid x \in A\} \cup \{1x \mid x \in B\}$.

We define the notion of computation path of oracle machines independent of any concrete oracle. A *computation path* of an oracle NPTM N encodes a complete valid computation that N can have relative to some/any oracle, i.e., it contains the sequence of configurations including the query strings and the answers from the oracle. Hence two computation paths ρ_1 and ρ_2 of an oracle NPTM are equal if and only if the configuration sequences, oracles queries, and oracles answers are the same for the computation paths. For any computation path ρ , let $Q^+(\rho)$ denote the set of strings that are queried along ρ and answered positively, and let $Q^-(\rho)$ denote the set of strings that are queried along ρ and answered negatively. Let $Q(\rho) = Q^+(\rho) \cup Q^-(\rho)$. For any concrete oracle A and input x , a given path ρ may or may not appear in $N^A(x)$. For instance, if $\alpha \in Q^+(\rho)$ then ρ does not appear in $N^A(x)$ for any A with $\alpha \notin A$. In this case we also say “ $N^A(x)$ does not have path ρ .”

For any complexity class \mathcal{C} and for any natural notion of polynomial-time reducibility r (e.g., $r \in \{m, dtt, tt, k-tt, T, k-T, b\}$), let $R_r^p(\mathcal{C})$ denote the closure of \mathcal{C} under r . That is, $R_r^p(\mathcal{C}) =_{df} \{L \mid (\exists L' \in \mathcal{C})[L \leq_r^p L']\}$. We refer the reader to any standard textbook in complexity theory (e.g. [BC93, HO02, Pap94]) for complexity classes and reductions not defined in this paper.

Given a complexity class \mathcal{C} , the unique existential ($\exists!$) and the unique universal ($\forall!$) operators on \mathcal{C} yield complexity classes. Formally:

Definition 2.1 *For any arbitrary complexity class \mathcal{C} ,*

1. $\exists! \cdot \mathcal{C}$ *is defined to be the class of all sets L for which there exists a polynomial $p(\cdot)$ and a set $L' \in \mathcal{C}$ such that for all $x \in \Sigma^*$,*

$$\begin{aligned} x \in L &\implies (\text{there exists a unique } y \in \Sigma^{p(|x|)})[\langle x, y \rangle \in L'], \text{ and} \\ x \notin L &\implies (\text{for all } y \in \Sigma^{p(|x|)})[\langle x, y \rangle \notin L']. \end{aligned}$$

2. $\forall! \cdot \mathcal{C}$ is defined to be the class of all sets L for which there exists a polynomial $p(\cdot)$ and a set $L' \in \mathcal{C}$ such that for all $x \in \Sigma^*$,

$$\begin{aligned} x \in L &\implies (\text{for all } y \in \Sigma^{p(|x|)})[\langle x, y \rangle \in L'], \text{ and} \\ x \notin L &\implies (\text{there exists a unique } y \in \Sigma^{p(|x|)})[\langle x, y \rangle \notin L']. \end{aligned}$$

We introduce the notion of a $\Sigma_k(A)$ -system. This notion is useful for concisely representing the computation of a stack of oracle NPTMs.

Definition 2.2 1. For any $k \in \mathbb{N}^+$ and $A \subseteq \Sigma^*$, we call a tuple $[A; N_1, N_2, \dots, N_k]$, where A is an oracle and N_1, N_2, \dots, N_k are nondeterministic oracle Turing machines, a $\Sigma_k(A)$ -system. The computation of a $\Sigma_k(A)$ -system $[A; N_1, N_2, \dots, N_k]$ on input x , denoted by $[A; N_1, N_2, \dots, N_k](x)$, is defined as follows:

- For $k = 1$, $[A; N_1](x) =_{df} N_1^A(x)$, and
- for $k > 1$, $[A; N_1, N_2, \dots, N_k](x) =_{df} N_1^{L(N_2^{L(N_3^{L(N_4^{L(N_5^{L(N_6^{L(N_7^{L(N_8^{L(N_1^A)} \dots))} \dots))} \dots))} \dots))} \dots)}(x)$.

2. The language accepted by a $\Sigma_k(A)$ -system, denoted by $L[A; N_1, N_2, \dots, N_k]$, is defined inductively as follows:

$$L[A; N_1, N_2, \dots, N_k] =_{df} \begin{cases} L(N_1^A) & \text{if } k = 1, \text{ and} \\ L(N_1^{L[A; N_2, N_3, \dots, N_k]}) & \text{if } k > 1. \end{cases}$$

We define the notion of unambiguity in $\Sigma_k(A)$ -systems as follows:

Definition 2.3 1. We say that a $\Sigma_k(A)$ -system $[A; N_1, N_2, \dots, N_k]$ is unambiguous if for every $1 \leq i \leq k$ and for every $x \in \Sigma^*$, $[A; N_i, N_{i+1}, \dots, N_k](x)$ has at most one accepting path.

2. For any $\Sigma_k(A)$ -system $[A; N_1, N_2, \dots, N_k]$, we define

$$L_{\text{unambiguous}}[A; N_1, N_2, \dots, N_k] = \begin{cases} L[A; N_1, N_2, \dots, N_k] & \text{if } [A; N_1, N_2, \dots, N_k] \text{ is} \\ & \text{unambiguous,} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Roughly speaking, a property of an oracle machine is called *robust* if the machine retains that property with respect to every oracle. Below we define the property of *robust unambiguity* for a $\Sigma_k(A)$ -system.

Definition 2.4 We say that a $\Sigma_k(A)$ -system $[A; N_1, N_2, \dots, N_k]$ is robustly unambiguous if for every set B , the $\Sigma_k(A \oplus B)$ -system $[A \oplus B; N_1, N_2, \dots, N_k]$ is unambiguous.

2.2 Promise Problems and Smart Reductions

Even, Selman, and Yacobi [ESY84] introduced and studied the notion of promise problems. Promise problems are generalizations of decision problems in that the set of Yes-instances and the set of No-instances must partition the set of all instances in a decision problem, whereas this is not necessarily the case with promise problems. Thus, for a promise problem a set of disallowed strings may be defined, which represent neither Yes-instances nor No-instances. Over the years, the notion of promise problems has proved to be useful at several places in computational complexity theory. (See [Gol05] for a nice survey on some applications of promise problems in computational complexity theory.)

Definition 2.5 (Based on [Gol05]; cf. [ESY84]) *A promise problem $\Pi = (\Pi_{\text{yes}}, \Pi_{\text{no}})$ is defined in terms of disjoint sets $\Pi_{\text{yes}}, \Pi_{\text{no}} \subseteq \Sigma^*$. The set Π_{yes} is called the set of Yes-instances, the set Π_{no} is called the set of No-instances, and the set $\Pi_{\text{yes}} \cup \Pi_{\text{no}}$ is called the promise set.*

Some technicalities are involved when oracle access to a promise problem is defined. If a query to a promise problem falls inside the promise set, then the answer to the query is well-defined (i.e., the answer is 1 if q is a Yes-instance and is 0 if q is a No-instance). However, if a query falls outside the promise set, then it is not immediately clear how that query should be handled by the promise problem, i.e., the oracle. Several natural models of oracle access to a promise problem are definable. (See [GS88, CHV93] for a few possible approaches to defining oracle accesses to promise problems.)

Grollmann and Selman [GS88] proposed a model of oracle access to a promise problem that *prohibits* queries that fall outside the promise set. In this model, a querying machine always asks queries from the promise set, i.e., the queries asked by the querying machine always obey the underlying promise of the promise problem. For instance, let us define a promise problem $\Pi_{\text{unique}} = (\Pi_{\text{yes}}, \Pi_{\text{no}})$ in terms of acceptance mechanism of a NPTM N as follows: $\Pi_{\text{yes}} = \{x \in \Sigma^* \mid \#\text{acc}_N(x) = 1\}$ and $\Pi_{\text{no}} = \{x \in \Sigma^* \mid \#\text{acc}_N(x) = 0\}$. Then a Turing access to Π_{unique} in the model proposed by Grollmann and Selman [GS88] requires that for any query y asked by the querying machine on some input, the computation of N on y must be unambiguous, i.e., $\#\text{acc}_N(y)$ must be either 0 or 1. A Turing reduction that obeys the constraints of this model (i.e., any query ever asked belongs to the promise set) is called a *smart* Turing reduction [GS88]. The definition given below formally captures the notion of smart Turing reduction from a decision problem to a promise problem.¹

Definition 2.6 *A set L polynomial-time smart Turing reduces to a promise problem $\Pi = (\Pi_{\text{yes}}, \Pi_{\text{no}})$, denoted by $L \leq_{s,T}^p \Pi$ or $L \in P_s^\Pi$, if there is a deterministic polynomial-time oracle Turing machine M such that for all $x \in \Sigma^*$,*

1. $x \in L \iff M^\Pi(x)$ accepts, and
2. if $M^\Pi(x)$ asks a query y to Π , then $y \in \Pi_{\text{yes}} \cup \Pi_{\text{no}}$.

¹Cai, Hemachandra, and Vyskoč [CHV93] referred to Grollmann and Selman's *smart* oracle access by the term *guarded* access.

If on all inputs $x \in \Sigma^*$, the querying machine M asks at most k queries, for some integer constant $k \geq 1$, then we say that L polynomial-time smart k -Turing reduces to Π and write $L \leq_{s,k-T}^p \Pi$ or $L \in P_s^{\Pi[k]}$.

In the above definition, we followed Grollmann and Selman's notion of smart Turing reductions from *decision* problems to *promise* problems. We may extend this notion to define reductions that reduce *promise* problems to *promise* problems. (See, for instance, [Gol05] for a generalization of smart Turing reductions to reductions among promise problems.) In this paper, we will only consider smart Turing reductions (i.e., reductions from decision problems to promise problems) as given by Grollmann and Selman.

The following two definitions are standard.

Definition 2.7 Let Π be any promise problem. $R_{s,T}^p(\Pi)$ is the class of all sets L such that $L \leq_{s,T}^p \Pi$; for all $k \in \mathbb{N}^+$, $R_{s,k-T}^p(\Pi)$ is the class of all sets L such that $L \leq_{s,k-T}^p \Pi$; $R_{s,b}^p(\Pi)$ is the class of all sets L for which there exists some $k \in \mathbb{N}^+$ such that $L \leq_{s,k-T}^p \Pi$.

Definition 2.8 For any class of promise problems \mathcal{C} and any reduction $r \in \{T, k-T, b\}$, we define $R_{s,r}^p(\mathcal{C}) =_{df} \bigcup_{\Pi \in \mathcal{C}} R_{s,r}^p(\Pi)$.

We will study the computational power of smart Turing reductions to a particular class of promise problems, namely the class Promise-UP, which is defined as follows.

Definition 2.9 Promise-UP is the class of all promise problems $\Pi = (\Pi_{\text{yes}}, \Pi_{\text{no}})$ for which there exists a nondeterministic polynomial-time Turing machine N such that for all $x \in \Sigma^*$,

$$\begin{aligned} x \in \Pi_{\text{yes}} &\implies \#\text{acc}_N(x) = 1, \text{ and} \\ x \in \Pi_{\text{no}} &\implies \#\text{acc}_N(x) = 0. \end{aligned}$$

The class $P_s^{\text{Promise-UP}}$ of sets that polynomial-time smart Turing reduce to Promise-UP is a prominent class that behaves remarkably differently than the related class P^{UP} . While $P_s^{\text{Promise-UP}}$ is known to contain the class FewP and the graph isomorphism problem [AK02], similar results for the case of P^{UP} are unknown.²

2.3 Unambiguity Based Hierarchies

Niedermeier and Rossmanith [NR98] observed that the notion of unambiguity in NPTMs can be generalized in three ways, each of which define an unambiguity based hierarchy.

Definition 2.10 (Unambiguity Based Hierarchies [LR94,NR98]) 1. The alternating unambiguous polynomial hierarchy is defined as:

$$\text{AUPH} =_{df} \bigcup_{k \geq 0} \text{A}\Sigma_k^p = \bigcup_{k \geq 0} \text{A}\Pi_k^p,$$

²Arvind and Kurur [AK02] showed that the graph isomorphism problem (GI) belongs to SPP, a class introduced in [Gup92,OH93,FFK94]. Subsequently, Crasmaru et al. [CGRS04] observed that the proof of classifying GI into SPP, as given by Arvind and Kurur [AK02], actually yields a somewhat improved classification for GI: GI belongs to $R_{s,T}^p(\text{Promise-UP})$, a subclass of SPP [CGRS04].

where

$$\text{AU}\Sigma_k^p = \begin{cases} \text{P} & \text{if } k = 0, \\ \exists! \cdot \text{AU}\Pi_{k-1}^p & \text{if } k \geq 1, \end{cases} \quad \text{and} \quad \text{AU}\Pi_k^p = \begin{cases} \text{P} & \text{if } k = 0, \\ \forall! \cdot \text{AU}\Sigma_{k-1}^p & \text{if } k \geq 1. \end{cases}$$

2. The unambiguous polynomial hierarchy is defined as:

$$\text{UPH} =_{df} \bigcup_{k \geq 0} \text{U}\Sigma_k^p = \bigcup_{k \geq 0} \text{U}\Pi_k^p,$$

where

$$\text{U}\Sigma_k^p = \begin{cases} \text{P} & \text{if } k = 0, \\ \text{UP}^{\text{U}\Sigma_{k-1}^p} & \text{if } k \geq 1, \end{cases} \quad \text{and} \quad \text{U}\Pi_k^p = \begin{cases} \text{P} & \text{if } k = 0, \\ \text{coUP}^{\text{U}\Sigma_{k-1}^p} & \text{if } k \geq 1. \end{cases}$$

3. The promise unambiguous polynomial hierarchy is defined as:

$$\mathcal{UPH} =_{df} \bigcup_{k \geq 0} \mathcal{U}\Sigma_k^p = \bigcup_{k \geq 0} \mathcal{U}\Pi_k^p,$$

where $\mathcal{U}\Sigma_0^p =_{df} \text{P}$, $\mathcal{U}\Sigma_1^p =_{df} \text{UP}$, and for every $k \geq 2$, $\mathcal{U}\Sigma_k^p$ is the class of all sets $L \in \Sigma_k^p$ such that for some oracle NPTMs N_1, N_2, \dots, N_k , $L = L(N_1^{L(N_2^{\dots^{L(N_k)})})})$, and for every $x \in \Sigma^*$ and for every $1 \leq i \leq k-1$, $N_1^{L(N_2^{\dots^{L(N_k)})}}(x)$ has at most one accepting path and if N_i asks a query w to its oracle $L(N_{i+1}^{\dots^{L(N_k)})}$ during the computation of $N_1^{\dots^{L(N_k)}}(x)$, then $N_{i+1}^{\dots^{L(N_k)}}(w)$ has at most one accepting path. For each $k \geq 0$, the class $\mathcal{U}\Pi_k^p$ is defined to be: $\mathcal{U}\Pi_k^p =_{df} \text{co}\mathcal{U}\Sigma_k^p$.

The following inclusion relationships between unambiguity based classes and other central classes are known (see also Figure 1).

Theorem 2.11 1. For all $k \geq 0$, $\text{AU}\Sigma_k^p \subseteq \text{U}\Sigma_k^p \subseteq \mathcal{U}\Sigma_k^p \subseteq \Sigma_k^p$ [LR94].

2. For all $k \geq 1$, $\text{UP}_{\leq k} \subseteq \text{AU}\Sigma_k^p \subseteq \text{U}\Sigma_k^p \subseteq \mathcal{U}\Sigma_k^p \subseteq \text{UAP} \subseteq \text{SPP}$ ([LR94] + [NR98] + [CGRS04]).

Despite the attention these hierarchies deserve, much less is known about the structure of these hierarchies since Lange and Rossmanith [LR94] first posed questions—such as, whether these hierarchies intertwine, or whether some unambiguity based hierarchy is contained in a fixed level of some other hierarchy, or whether some/all of these hierarchies collapse to a fixed level—on their structure. On the positive side, there had been some advances in understanding the structure of these hierarchies. Hemaspaandra and Rothe [HR97] related the structure of these hierarchies to the existence of sparse Turing complete sets for UP. The structure of these hierarchies received renewed interests in some recent works (see [ACRW04,

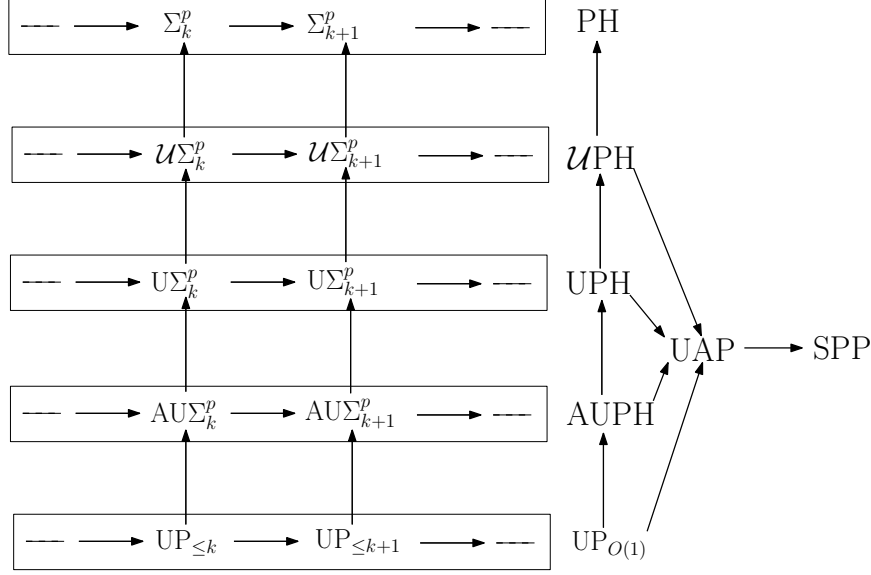


Figure 1: Known inclusion structure of unambiguity based classes and other central classes. The arrows point from subclasses to superclasses.

CGRS04,ST,GT05]). In particular, Spakowski and Tripathi [ST] investigated the relativized structure of these hierarchies. They proved that the unambiguity based hierarchies AUPH, UPH, and \mathcal{UPH} are infinite in some relativized world. They also proved a contrasting result on their relativized structure: For each $k \geq 2$, there is a relativized world where these hierarchies collapse so that they have exactly k distinct levels and their k 'th levels coincide with PSPACE.

3 Proof Technique

3.1 Main Lemma

Our main lemma is Lemma 3.1, which we will use throughout this paper for generalizing known oracle constructions involving unambiguity based classes such as UP and Promise-UP to new oracle constructions involving arbitrary levels of the UPH. Roughly, Lemma 3.1 states computational limitations of a $\Sigma_k(\mathcal{O})$ -system, for any arbitrary $k \geq 1$, under certain weak conditions.

Lemma 3.1 *Fix a $\Sigma_k(\mathcal{O})$ -system $[\mathcal{O}; N_1, N_2, \dots, N_k]$, a string $x \in \Sigma^*$, and a set $U \subseteq \Sigma^*$ such that $\mathcal{O} \cap U = \emptyset$. Let $r(\cdot)$ be a polynomial that bounds the running time of each of the machines N_1, N_2, \dots, N_k . Then the following holds:*

1. Suppose $[\mathcal{O}; N_1, N_2, \dots, N_k](x)$ accepts and for every $A \subseteq U$ with $\|A\| \leq k$, $[\mathcal{O} \cup$

$A; N_1, N_2, \dots, N_k]$ is unambiguous. Let

$$C = \{\alpha \in U \mid [\mathcal{O} \cup \{\alpha\}; N_1, N_2, \dots, N_k](x) \text{ rejects}\}.$$

Then $\|C\| \leq 5^k \cdot \prod_{i=1}^k (r \circ)^i(|x|)$.

2. Suppose $[\mathcal{O}; N_1, N_2, \dots, N_k](x)$ rejects and for every $A \subseteq U$ with $\|A\| \leq k+1$, $[\mathcal{O} \cup A; N_1, N_2, \dots, N_k]$ is unambiguous. Let

$$C = \{\alpha \in U \mid [\mathcal{O} \cup \{\alpha\}; N_1, N_2, \dots, N_k](x) \text{ accepts}\}.$$

Then $\|C\| \leq 5^k \cdot \prod_{i=1}^k (r \circ)^i(|x|)$.

Proof We prove (1) and (2) by induction on k . For the base case $k = 1$ of (1), we have $[\mathcal{O}; N_1](x)$ accepts. Also since $[\mathcal{O}; N_1]$ is unambiguous by the assumption made in (1), there is a unique accepting path in $[\mathcal{O}; N_1](x)$. Let C' be the set of all queries $w \in U$ along this unique accepting path. Then clearly $\|C\| \leq \|C'\| \leq r(|x|)$. Thus (1) holds for the base case.

For the base case $k = 1$ of (2), we have $[\mathcal{O}; N_1](x)$ rejects. Suppose that $\|C\| > 5 \cdot r(|x|)$. Note that for every $\alpha \in C$, $[\mathcal{O} \cup \{\alpha\}; N_1](x)$ accepts. Thus, for every $\alpha \in C$, let $\lambda(\alpha)$ be an accepting path in $[\mathcal{O} \cup \{\alpha\}; N_1](x)$. It is easy to show that $\lambda(\alpha_1) \neq \lambda(\alpha_2)$ for any distinct $\alpha_1, \alpha_2 \in C$. To see this, let $\rho = \lambda(\alpha)$ for some $\alpha \in C$. Then, by the definition of $\lambda(\alpha)$ and the assumption that $[\mathcal{O}; N_1](x)$ rejects, we notice that path ρ appears in $N_1^{\mathcal{O} \cup \{\alpha\}}(x)$ but does not appear in $N_1^{\mathcal{O}}(x)$. From this, it follows that α must be answered positively along ρ , i.e., $\alpha \in Q^+(\rho)$, since otherwise ρ would also appear in $N_1^{\mathcal{O}}(x)$. Therefore, for any oracle \mathcal{B} with $\alpha \notin \mathcal{B}$, ρ cannot appear in $N_1^{\mathcal{B}}(x)$. In particular, ρ cannot appear in $N_1^{\mathcal{O} \cup \{\alpha'\}}(x)$ for any α' with $\alpha' \neq \alpha$. Hence, we get that $\lambda(\alpha_1) \neq \lambda(\alpha_2)$ for any distinct $\alpha_1, \alpha_2 \in C$.

We define for any $\alpha \in C$,

$$\text{conflicting}(\alpha) = \{\beta \in C \mid \lambda(\alpha) \text{ is not an accepting path in } [\mathcal{O} \cup \{\alpha, \beta\}; N_1]\}.$$

Since $N_1(x)$ with any oracle asks at most $r(|x|)$ queries, there can be at most $r(|x|)$ strings $\beta \in C$ that can cause $\lambda(\alpha)$ not to appear in $[\mathcal{O} \cup \{\alpha, \beta\}; N_1]$. In other words, for any $\alpha \in C$, it holds that $\|\text{conflicting}(\alpha)\| \leq r(|x|)$. Thus, it follows by an easy counting argument and the assumption that $\|C\| > 5 \cdot r(|x|)$ that there exist distinct $\alpha_1, \alpha_2 \in C$ such that $\alpha_1 \notin \text{conflicting}(\alpha_2)$ and $\alpha_2 \notin \text{conflicting}(\alpha_1)$. We had already shown that $\lambda(\alpha_1)$ and $\lambda(\alpha_2)$ are distinct paths for distinct $\alpha_1, \alpha_2 \in C$. Therefore, $[\mathcal{O} \cup \{\alpha_1, \alpha_2\}; N_1](x)$ has two distinct accepting paths, namely $\lambda(\alpha_1)$ and $\lambda(\alpha_2)$. This contradicts our assumption that $[\mathcal{O} \cup A; N_1]$ is unambiguous for every $A \subseteq U$ with $\|A\| \leq 2$. Thus also (2) holds for the base case.

Induction Hypothesis: Assume that (1) and (2) in the statement of Lemma 3.1 hold for some $k \geq 1$.

Inductive Step: Let $[\mathcal{O}; N_1, N_2, \dots, N_{k+1}]$ be a $\Sigma_{k+1}(\mathcal{O})$ -system and let $r(\cdot)$ be a polynomial that bounds the running time of each of $N_1, N_2, \dots, N_k, N_{k+1}$.

We first prove (1). Suppose $[\mathcal{O}; N_1, N_2, \dots, N_{k+1}](x)$ accepts and $[\mathcal{O} \cup A; N_1, N_2, \dots, N_{k+1}]$ is unambiguous for every $A \subseteq U$ with $\|A\| \leq k+1$. Let λ denote the unique accepting path of $[\mathcal{O}; N_1, N_2, \dots, N_{k+1}](x)$. For every query w along λ , the $\Sigma_k(\mathcal{O})$ -system $[\mathcal{O}; N_2, N_3, \dots, N_{k+1}]$ computes $[\mathcal{O}; N_2, N_3, \dots, N_{k+1}](w)$. By Definition 2.3(1), for every $A \subseteq U$ with $\|A\| \leq k+1$, the $\Sigma_k(A)$ -system $[\mathcal{O} \cup A; N_2, N_3, \dots, N_{k+1}]$ is unambiguous. Thus, it follows by the induction hypothesis that for every query w along λ and for all but at most $5^k \cdot \prod_{i=1}^k (r \circ)^i(\|w\|) \leq 5^k \cdot \prod_{i=2}^{k+1} (r \circ)^i(\|x\|)$ strings $\alpha \in U$, adding $\alpha \in U$ to \mathcal{O} does not change the decision (i.e., acceptance or rejection) of $[\mathcal{O}; N_2, N_3, \dots, N_{k+1}](w)$. Since the number of such queries w along λ is at most $r(\|x\|)$, it follows that λ is an accepting path in $[\mathcal{O} \cup \{\alpha\}; N_1, N_2, \dots, N_{k+1}](x)$ for all but at most $r(\|x\|) \cdot 5^k \cdot \prod_{i=2}^{k+1} (r \circ)^i(\|x\|) < 5^{k+1} \cdot \prod_{i=1}^{k+1} (r \circ)^i(\|x\|)$ strings $\alpha \in U$. This proves (1), the first part of the inductive step.

We now prove (2). Suppose that $\|C\| > 5^{k+1} \cdot \prod_{i=1}^{k+1} (r \circ)^i(\|x\|)$. For any $\alpha \in C$, let $\lambda(\alpha)$ denote the unique accepting path in $[\mathcal{O} \cup \{\alpha\}; N_1, N_2, \dots, N_{k+1}](x)$. We define an equivalence relation σ on C as follows: For every $\alpha_1, \alpha_2 \in C$,

$$\alpha_1 \sigma \alpha_2 \iff \lambda(\alpha_1) = \lambda(\alpha_2).$$

The following cases are exhaustive.

Case 1: There is an equivalence class of σ of size $> 5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(\|x\|)$. Let $[\alpha]$ be such an equivalence class. Then we have the following situation: The accepting path $\lambda(\alpha)$ does not appear in $[\mathcal{O}; N_1, N_2, \dots, N_{k+1}](x)$, but for every $\beta \in [\alpha]$, path $\lambda(\alpha)$ appears in $[\mathcal{O} \cup \{\beta\}; N_1, N_2, \dots, N_{k+1}](x)$. Hence for every query $w \in \Sigma^*$ along $\lambda(\alpha)$, it holds that for every $\beta, \beta' \in [\alpha]$, $[\mathcal{O} \cup \{\beta\}; N_2, N_3, \dots, N_{k+1}](w)$ and $[\mathcal{O} \cup \{\beta'\}; N_2, N_3, \dots, N_{k+1}](w)$ have the same acceptance behavior, i.e., $[\mathcal{O} \cup \{\beta\}; N_2, N_3, \dots, N_{k+1}](w)$ accepts if and only if $[\mathcal{O} \cup \{\beta'\}; N_2, N_3, \dots, N_{k+1}](w)$ accepts. There must be at least one query $w' \in \Sigma^*$ along $\lambda(\alpha)$ such that for some (and hence by the previous sentence for *every*) $\beta \in [\alpha]$, adding β to \mathcal{O} changes the answer of $[\mathcal{O}; N_2, N_3, \dots, N_{k+1}](w')$, since otherwise $\lambda(\alpha)$ would also appear in $[\mathcal{O}; N_1, N_2, \dots, N_{k+1}](x)$. Also note that for every $A \subseteq U$ with $\|A\| \leq k+1$, $[\mathcal{O} \cup A; N_2, N_3, \dots, N_{k+1}]$ is unambiguous. Thus we get a contradiction with the induction hypothesis, since $\|[\alpha]\| > 5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(\|x\|) \geq 5^k \cdot \prod_{i=2}^{k+1} (r \circ)^i(\|x\|) \geq 5^k \cdot \prod_{i=1}^k (r \circ)^i(\|w'\|)$.

Case 2: Every equivalence class of σ is of size $\leq 5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(\|x\|)$. We need the following claim, which demonstrates that if σ is an equivalence relation over some set C consisting only of small equivalence classes, then C can be partitioned into two sufficiently large disjoint sets C_1 and C_2 such that every equivalence class is contained in either C_1 or C_2 .

Claim 1 *Let σ be any equivalence relation over some set C and let $\|C\| > 5s$. If $\|[\alpha]\| \leq s$ for every $\alpha \in C$, then there exists a set $J \subseteq C$ such that $2s < \|\bigcup_{\alpha \in J} [\alpha]\| < \|C\| - 2s$.*

Proof of Claim 1. Take any set $J' \subseteq C$ of maximum cardinality such that $\|\bigcup_{\alpha \in J'} [\alpha]\| \leq 2s$. Let $J = J' \cup [\beta]$, where β is any element in $C - J'$ (note that β exists because

$\|C\| > 2s$). Set J witnesses the correctness of the claim, which can be seen as follows. First, $\|\bigcup_{\alpha \in J} [\alpha]\| > 2s$ because $\|J\| > \|J'\|$ and J' is a maximum cardinality set such that $\|\bigcup_{\alpha \in J'} [\alpha]\| \leq 2s$. Second, $\|\bigcup_{\alpha \in J} [\alpha]\| \leq 2s + s = 3s$ because $\|\bigcup_{\alpha \in J'} [\alpha]\| \leq 2s$ and $\|[\beta]\| \leq s$. Since $\|C\| > 5s$, this implies that $\|\bigcup_{\alpha \in J} [\alpha]\| < \|C\| - 2s$. \blacksquare (Claim 1)

It is easy to see that our equivalence class σ over C satisfies the preconditions of Claim 1. Hence, it follows that there exists a set $J \subseteq C$ such that

$$2 \cdot 5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(|x|) < \left\| \bigcup_{\alpha \in J} [\alpha] \right\| < \|C\| - 2 \cdot 5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(|x|).$$

Let $C_1 := \bigcup_{\alpha \in J} [\alpha]$ and let $C_2 := C - C_1$. Then both $\|C_1\|$ and $\|C_2\|$ are greater than $2 \cdot 5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(|x|)$. For every $\alpha_1 \in C_1$ and $\alpha_2 \in C_2$, we define $\text{conflicting}(\alpha_1) =$

$$\{\beta_2 \in C_2 \mid \lambda(\alpha_1) \text{ is not an accepting path in } [\mathcal{O} \cup \{\alpha_1, \beta_2\}; N_1, N_2, \dots, N_{k+1}](x)\},$$

and define $\text{conflicting}(\alpha_2) =$

$$\{\beta_1 \in C_1 \mid \lambda(\alpha_2) \text{ is not an accepting path in } [\mathcal{O} \cup \{\alpha_2, \beta_1\}; N_1, N_2, \dots, N_{k+1}](x)\}.$$

We claim that both $\|\text{conflicting}(\alpha_1)\|$ and $\|\text{conflicting}(\alpha_2)\|$ are no more than $5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(|x|)$. To prove this, first note that $N_1(x)$ with any oracle can ask at most $r(|x|)$ queries along a computation path. Second, by the induction hypothesis, for every query w made by $N_1(x)$, there can be at most $5^k \cdot \prod_{i=1}^k (r \circ)^i(|w|) \leq 5^k \cdot \prod_{i=2}^{k+1} (r \circ)^i(|x|)$ strings $\beta \in U$ such that adding β to some oracle \mathcal{O}' changes the decision (acceptance or rejection) of $[\mathcal{O}'; N_2, N_3, \dots, N_{k+1}](w)$. These two facts imply the stated bound on $\|\text{conflicting}(\alpha_1)\|$ and $\|\text{conflicting}(\alpha_2)\|$.

With the bounds $\|C_1\|, \|C_2\| > 2 \cdot 5^k \cdot \prod_{i=1}^{k+1} (r \circ)^i(|x|)$, a simple counting argument now shows that there exist $\alpha_1 \in C_1$ and $\alpha_2 \in C_2$ such that $\alpha_1 \notin \text{conflicting}(\alpha_2)$ and $\alpha_2 \notin \text{conflicting}(\alpha_1)$. As a consequence, $[\mathcal{O} \cup \{\alpha_1, \alpha_2\}; N_1, N_2, \dots, N_{k+1}](x)$ accepts along two distinct paths, namely $\lambda(\alpha_1)$ and $\lambda(\alpha_2)$, which are indeed distinct since $[\alpha_1] \neq [\alpha_2]$. This gives a contradiction with the assumption that $[\mathcal{O} \cup A; N_1, N_2, \dots, N_{k+1}]$ is unambiguous for every $A \subseteq U$ with $\|A\| \leq k+1$. \blacksquare (Lemma 3.1)

3.2 The Notion of (h, t) -ambiguity for Functions on $\wp(\Sigma^*)$

Any oracle machine can be interpreted as a function mapping a set of strings to another set of strings as follows: A machine N maps any set \mathcal{O} to the set $L(N^\mathcal{O})$. Therefore it makes sense to consider the (possibly partial) function $\mathcal{L} : \wp(\Sigma^*) \rightarrow \wp(\Sigma^*)$ defined by a $\Sigma_k(\cdot)$ -system $[\cdot; N_1, N_2, \dots, N_k]$. (That is, define \mathcal{L} so that for every $\mathcal{O} \subseteq \Sigma^*$, $\mathcal{L}(\mathcal{O}) =_{df} L_{\text{unambiguous}}[\mathcal{O}; N_1, N_2, \dots, N_k]$.) We introduce a convenient notion called “ (h, t) -ambiguity” for (partial or total) functions, which we will later apply to functions defined by $\Sigma_k(\cdot)$ -systems.

Definition 3.2 For any $h \in \mathbb{N}^+$ and polynomial $t(\cdot)$, we call a partial or total function $\mathcal{L} : \wp(\Sigma^*) \rightarrow \wp(\Sigma^*)$ (h, t) -ambiguous if for every $\mathcal{O}, U \subseteq \Sigma^*$ with $\mathcal{O} \cap U = \emptyset$, one of the following is true:

1. For some $A \subseteq U$ with $\|A\| \leq h$, $\mathcal{L}(\mathcal{O} \cup A)$ is undefined, or
2. for every $w \in \Sigma^*$,

$$\|\{\alpha \in U \mid w \in \mathcal{L}(\mathcal{O} \cup \{\alpha\}) \iff w \notin \mathcal{L}(\mathcal{O})\}\| \leq t(|w|).$$

The machine N_1 in a $\Sigma_k(\mathcal{O})$ -system $[\mathcal{O}; N_1, N_2, \dots, N_k]$ has oracle access to the set $L[\mathcal{O}; N_2, N_3, \dots, N_k]$. In many of our proofs, we first apply Lemma 3.1 to prove that under certain conditions the $\Sigma_{k-1}(\cdot)$ -subsystem $[\cdot; N_2, N_3, \dots, N_k]$ defines a (k, t) -ambiguous function \mathcal{L}' , where t is some polynomial and for any $\mathcal{O} \subseteq \Sigma^*$, $\mathcal{L}'(\mathcal{O})$ is defined to be $L[\mathcal{O}; N_2, N_3, \dots, N_k]$. The (k, t) -ambiguity of the function \mathcal{L}' defined by the $\Sigma_{k-1}(\cdot)$ -subsystem $[\cdot; N_2, N_3, \dots, N_k]$ is the only property of \mathcal{L}' that is needed in our proofs. Therefore, we can assume without loss of generality that in a $\Sigma_k(\mathcal{O})$ -system $[\mathcal{O}; N_1, N_2, \dots, N_k]$, machine N_1 has oracle access to the set $\mathcal{L}(\mathcal{O})$, where \mathcal{L} can be any arbitrary (k, t) -ambiguous function, rather than to the set $L[\mathcal{O}; N_2, N_3, \dots, N_k]$. Using this approach has its advantages: It greatly simplifies our proof arguments and makes expressions compact, since we no longer need to deal with stacks of oracle NPTMs.

4 Applications

In this section, we demonstrate applications of our proof techniques. We show that our counting techniques are useful in generalizing certain known relativization results involving bounded ambiguity classes such as UP and Promise-UP to new results involving arbitrary levels of UPH. This stands in contrast to generalizations achieved for relativization results involving levels of PH. For instance, extending the relativized separation of initial levels of PH [BGS75, BS79] to relativized separations of arbitrary levels of PH [Yao85] required applications of sophisticated circuit-theoretic techniques. For the case of the unambiguous polynomial hierarchy however, we show (via Theorem 4.1) that a relativized separation of its levels can be achieved by counting techniques alone, i.e., without resorting to circuit-theoretic tools and techniques.

4.1 Comparing Bounded Ambiguity Classes with the Levels of UPH

We compare classes defined by nondeterministic polynomial-time Turing machines having restrictions on the number of accepting paths ($\text{UP}_{O(1)}$ and FewP) with levels of the unambiguous polynomial hierarchy (UPH). It is known that $\text{UP}_{\leq k} \subseteq \Sigma_k^P$ in all relativized worlds. Theorem 4.1 shows the optimality of this inclusion with respect to relativizable proof techniques. Beigel [Bei89] constructed an oracle relative to which $\text{UP}_{k(n)+1} \not\subseteq \text{UP}_{k(n)}$, for every polynomial $k(n) \geq 2$. Theorem 4.1 subsumes this oracle result of Beigel [Bei89] for any constant k .

By a slight modification of the oracle construction in Theorem 4.1, we can show that the second level of the promise unambiguous hierarchy $\mathcal{U}\Sigma_2^p$ is not contained in the unambiguous polynomial hierarchy UPH. Results on relativized separations of levels of some unambiguity based hierarchy from another hierarchy have been investigated earlier. Rossmanith (see [NR98]) gave a relativized separation of $\text{AU}\Sigma_k^p$ from $\text{U}\Sigma_k^p$, for any $k \geq 2$. Spakowski and Tripathi [ST] constructed an oracle relative to which $\text{AU}\Sigma_k^p \not\subseteq \Pi_k^p$, for any $k \geq 1$. Our relativized separation of $\mathcal{U}\Sigma_2^p$ from UPH does not seem to be implied from these previous results in any obvious way.

Theorem 4.1 *For any integer $k \geq 1$, there exists an oracle \mathcal{A} such that $\text{UP}_{\leq k+1}^{\mathcal{A}} \not\subseteq \text{U}\Sigma_k^{p,\mathcal{A}}$.*

Proof Our test language is: $L(\mathcal{A}) = \{0^n \mid \mathcal{A}^{=n} \neq \emptyset\}$. We will create an oracle \mathcal{A} that, for any length $n \in \mathbb{N}^+$, satisfies $|\mathcal{A}^{=n}| \leq k+1$. Let $(N_{i,1}, N_{i,2}, \dots, N_{i,k})$ be an enumeration of tuples, where $N_{i,\star}$ is a nondeterministic polynomial-time oracle Turing machine with running time bounded by some polynomial $p(\cdot)$. Initially, $\mathcal{A} := \emptyset$.

Stage i : Choose a very large integer n such that (a) $2^n > 5^k \cdot \prod_{j=1}^k (p \circ)^j(n)$, (b) n satisfies the promises made in the previous stages, (c) no string of length $\geq n$ is ever queried in any of the previous stages, and (d) n is larger than the value in the previous stage.

If there exists a set $B \subseteq \Sigma^n$ such that $|B| \leq k+1$ and $[\mathcal{A} \cup B; N_{i,1}, N_{i,2}, \dots, N_{i,k}]$ is not unambiguous, then set $\mathcal{A} := \mathcal{A} \cup B$. Promise to choose the value of n in the next stage to be larger than $(p \circ)^k(|w|)$, where w is an arbitrary string witnessing that $[\mathcal{A} \cup B; N_{i,1}, N_{i,2}, \dots, N_{i,k}]$ is not unambiguous, and then move to the next stage.

Otherwise, choose a string $\alpha \in \Sigma^n$ (as guaranteed by Lemma 3.1 and by our choice of n) such that

$$[\mathcal{A}; N_{i,1}, N_{i,2}, \dots, N_{i,k}](0^n) \text{ accepts} \iff [\mathcal{A} \cup \{\alpha\}; N_{i,1}, N_{i,2}, \dots, N_{i,k}](0^n) \text{ accepts}.$$

If $[\mathcal{A}; N_{i,1}, N_{i,2}, \dots, N_{i,k}](0^n)$ accepts, then move to the next stage. Otherwise, if $[\mathcal{A}; N_{i,1}, N_{i,2}, \dots, N_{i,k}](0^n)$ rejects, then set $\mathcal{A} := \mathcal{A} \cup \{\alpha\}$ and move to the next stage.

End of Stage

Clearly the construction guarantees that $L(\mathcal{A}) \in \text{UP}_{\leq k+1}^{\mathcal{A}} - \text{U}\Sigma_k^{p,\mathcal{A}}$. ■

A straightforward adaptation of the proof of Theorem 4.1 allows to separate the second level, $\mathcal{U}\Sigma_2^p$, of the promise unambiguous polynomial hierarchy from the unambiguous polynomial hierarchy, UPH, in some relativized world. We obtain this relativized separation via Theorem 4.2, where the subclass $\text{FewP}^{\mathcal{A}}$ of $\mathcal{U}\Sigma_2^{p,\mathcal{A}}$ is separated from $\text{UPH}^{\mathcal{A}}$.

Theorem 4.2 *There exists an oracle \mathcal{A} such that $\text{FewP}^{\mathcal{A}} \not\subseteq \text{UPH}^{\mathcal{A}}$.*

Proof Sketch: Take the test language $L(\mathcal{A})$ used in the proof of Theorem 4.1. Maintain the stipulation that the oracle \mathcal{A} satisfies, for all $n \in \mathbb{N}^+$, $|\mathcal{A}^{=n}| \leq n$; this ensures that $L(\mathcal{A}) \in \text{FewP}^{\mathcal{A}}$. Finally, for all $k \in \mathbb{N}^+$, diagonalize against all tuples $(N_{i,1}, N_{i,2}, \dots, N_{i,k})$ as in the proof of Theorem 4.1. ■

Corollary 4.3 *There is a relativized world where $\mathcal{U}\Sigma_2^p$ is not contained in UPH.*

Cai, Hemachandra, and Vyskoč [CHV93] proved that smart 2-Turing access to Promise-UP cannot be subsumed by $\text{coNP}^{\text{UP}} \cup \text{NP}^{\text{UP}}$ in some relativized world. As a consequence, they showed that there is a relativized world where smart bounded adaptive reductions to Promise-UP and smart bounded nonadaptive reductions to Promise-UP are nonequivalent, a characteristic that stands in contrast with the cases of UP and NP. (Both UP and NP are known to have equivalence between bounded adaptive reductions and bounded nonadaptive reductions in all relativized worlds (see [CHV93, Wag90].) We obtain a generalization of their result as a corollary of Theorem 4.4: There is a relativized world where smart k -Turing access to Promise-UP is not contained in $\text{coNP}^{\text{UP}_{k-1}^{p, \mathcal{A}}} \cup \text{NP}^{\text{UP}_{k-1}^{p, \mathcal{A}}}$, for any $k \geq 2$. The proof of Theorem 4.4 gives a first example for the role of (h, t) -ambiguity in derivations of our results.

Theorem 4.4 *For any integer $k \geq 2$, there exists an oracle \mathcal{A} such that*

$$\text{UP}_{\leq k}^{\mathcal{A}} \not\subseteq \text{coNP}^{\text{UP}_{k-1}^{p, \mathcal{A}}}$$

Proof The test language $L(\mathcal{A})$ is the same as the one in the proof of Theorem 4.1. We maintain the stipulation that for every $n \in \mathbb{N}^+$, $\|\mathcal{A}^n\| \leq k$; then, clearly $L(\mathcal{A}) \in \text{UP}_{\leq k}^{\mathcal{A}}$. We will show that $L(\mathcal{A}) \notin \text{coNP}^{\text{UP}_{k-1}^{p, \mathcal{A}}}$.

Let $(N_{i,1}, N_{i,2}, \dots, N_{i,k})$ be an enumeration of tuples, where $N_{i,\star}$ is a nondeterministic polynomial-time oracle Turing machine. Initially, $\mathcal{A} := \emptyset$.

Stage i : Let $p(\cdot)$ be a polynomial that bounds the running time of $N_{i,\star}$. Choose a large integer n such that (a) $2^n > 5^{k-1} \cdot \prod_{i=1}^k (p\circ)^i(n)$, (b) n satisfies any promises made in the previous stages, (c) n is larger than the value of n in the previous stages, and (d) no queries of length $\geq n$ are made in the previous stages.

If there exists a set $B \subseteq \Sigma^n$ such that $\|B\| \leq k$ and $[\mathcal{A} \cup B; N_{i,2}, N_{i,3}, \dots, N_{i,k}]$ is not unambiguous, then set $\mathcal{A} := \mathcal{A} \cup B$. Promise to choose the value of n in the next stage to be larger than $(p\circ)^{k-1}(|w|)$, where w is any arbitrary string witnessing that $[\mathcal{A} \cup B; N_{i,2}, N_{i,3}, \dots, N_{i,k}]$ is not unambiguous, and then move to the next stage.

Otherwise proceed as follows. Let $\mathcal{L} : \wp(\Sigma^*) \rightarrow \wp(\Sigma^*)$ be the function defined so that for every $\mathcal{O} \subseteq \Sigma^*$,

$$\mathcal{L}(\mathcal{O}) =_{df} L_{\text{unambiguous}}[\mathcal{O}; N_{i,2}, N_{i,3}, \dots, N_{i,k}].$$

It follows from Lemma 3.1 that \mathcal{L} is (k, t) -ambiguous for some polynomial $t(\cdot) =_{df} 5^{k-1} \cdot \prod_{i=1}^{k-1} (p\circ)^i(\cdot)$. Note that $\mathcal{L}(\mathcal{A} \cup B)$ is defined for every $B \subseteq \Sigma^n$ satisfying $\|B\| \leq k$.

If $N_{i,1}^{\mathcal{L}(\mathcal{A})}(0^n)$ rejects, then move to the next stage. Otherwise, fix an accepting path ρ in $N_{i,1}^{\mathcal{L}(\mathcal{A})}(0^n)$. For each query $w \in Q(\rho)$, let $C(w) =_{df} \{\alpha \in \Sigma^n \mid w \in \mathcal{L}(\mathcal{A} \cup \{\alpha\}) \iff w \notin \mathcal{L}(\mathcal{A})\}$. By the definition of (k, t) -ambiguity, for each $w \in Q(\rho)$, we have $\|C(w)\| \leq t(|w|) \leq t(p(n))$. Because n is large enough, we can choose some $\alpha \in \Sigma^n$ such that $\alpha \notin \bigcup_{w \in Q(\rho)} C(w)$. Set $\mathcal{A} := \mathcal{A} \cup \{\alpha\}$ and move to the next stage.

End of Stage

Clearly the construction guarantees that $L(\mathcal{A}) \notin \text{coNP}^{\text{US}_{k-1}^{p,A}}$. ■

Corollary 4.5 *For any integer $k \geq 2$, there exists an oracle \mathcal{A} such that*

$$R_{s,k-T}^p(\text{Promise-UP}^{\mathcal{A}}) \not\subseteq \text{coNP}^{\text{US}_{k-1}^{p,A}} \cup \text{NP}^{\text{US}_{k-1}^{p,A}}.$$

Proof This follows from Theorem 4.4 because for all oracles \mathcal{A} , $\text{UP}_{\leq k}^{\mathcal{A}} \subseteq R_{s,k-T}^p(\text{Promise-UP}^{\mathcal{A}})$ and $R_{s,k-T}^p(\text{Promise-UP}^{\mathcal{A}})$ is closed under complementation. ■

4.2 Simulating Nonadaptive Access by Adaptive Access (Non-promise Case)

It is known that adaptive Turing access to NP is exponentially more powerful compared to nonadaptive Turing access to NP. That is, $R_{(2^k-1)-tt}^p(\text{NP}) \subseteq R_{k-T}^p(\text{NP})$ [Bei91] and this inclusion relativizes. However, for the case of unambiguous nondeterministic computation such a relationship between nonadaptive access and adaptive access is not known. Cai, Hemachandra, and Vyskoč [CHV92] showed that even proving the superiority of adaptive Turing access over nonadaptive Turing access with one single query more might be nontrivial for unambiguous nondeterministic computation:

Theorem 4.6 ([CHV92]) *For any total, polynomial-time computable and polynomially bounded function $k(\cdot)$, there exists an oracle \mathcal{A} such that*

$$R_{(k(n)+1)-tt}^p(\text{UP}^{\mathcal{A}}) \not\subseteq R_{k(n)-T}^{p,\mathcal{A}}(\text{UP}^{\mathcal{A}}).$$

In the next theorem, we generalize this result to the higher levels of the unambiguous polynomial hierarchy UPH.

Theorem 4.7 *For any total, polynomial-time computable and polynomially bounded function $k(\cdot)$, and integer $h \geq 1$, there exists an oracle \mathcal{A} such that*

$$R_{(k(n)+1)-dtt}^p(\text{UP}_{\leq h}^{\mathcal{A}}) \not\subseteq R_{k(n)-T}^{p,\mathcal{A}}(\text{US}_h^{p,\mathcal{A}}),$$

and hence $R_{(k(n)+1)-dtt}^p(\text{US}_h^{p,\mathcal{A}}) \not\subseteq R_{k(n)-T}^{p,\mathcal{A}}(\text{US}_h^{p,\mathcal{A}})$.

For the proof of Theorem 4.7, we need the following lemma.

Lemma 4.8 *Fix an oracle NPTM N with running time bounded by some polynomial $p(\cdot)$, string $x \in \Sigma^*$, and sets $\mathcal{O}, U_1, U_2 \subseteq \Sigma^*$ such that $\mathcal{O} \cap U_1 = \mathcal{O} \cap U_2 = U_1 \cap U_2 = \emptyset$. Let $\mathcal{L} : \wp(\Sigma^*) \rightarrow \wp(\Sigma^*)$ be an (h, t) -ambiguous function such that $\mathcal{L}(\mathcal{O} \cup A_1 \cup A_2)$ is defined for every $A_1 \subseteq U_1$ and $A_2 \subseteq U_2$ with $\|A_1\| \leq h$ and $\|A_2\| \leq h$. Let*

$$\begin{aligned} C_1 &= \{\alpha \in U_1 \mid N^{\mathcal{L}(\mathcal{O})}(x) \text{ accepts} \iff N^{\mathcal{L}(\mathcal{O} \cup \{\alpha\})}(x) \text{ rejects}\} \text{ and} \\ C_2 &= \{\alpha \in U_2 \mid N^{\mathcal{L}(\mathcal{O})}(x) \text{ accepts} \iff N^{\mathcal{L}(\mathcal{O} \cup \{\alpha\})}(x) \text{ rejects}\}. \end{aligned}$$

If $N^{\mathcal{L}(\mathcal{O} \cup A_1 \cup A_2)}(x)$ is unambiguous for every $A_1 \subseteq U_1$ and $A_2 \subseteq U_2$ with $\|A_1\| \leq 1$ and $\|A_2\| \leq 1$, then $\min\{\|C_1\|, \|C_2\|\} \leq 2 \cdot p(|x|) \cdot t(p(|x|)) \cdot (p(|x|) \cdot t(p(|x|)) + 1)$.

Proof We start with the easier case that $N^{\mathcal{L}(\mathcal{O})}(x)$ accepts. Let ρ be the (unique) accepting computation path in $N^{\mathcal{L}(\mathcal{O})}(x)$. Then $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha\})}(x)$ accepts unless for some query $w \in Q(\rho)$, it is the case that $w \in \mathcal{L}(\mathcal{O} \cup \{\alpha\}) \iff w \notin \mathcal{L}(\mathcal{O})$. Since $N^{\mathcal{L}(\mathcal{O})}(x)$ queries at most $p(|x|)$ queries along every path, since each query $w \in Q(\rho)$ is of length $\leq p(|x|)$, since \mathcal{L} is (h, t) -ambiguous, and since $\mathcal{L}(\mathcal{O} \cup A_1 \cup A_2)$ is defined for every $A_1 \subseteq U_1$ and $A_2 \subseteq U_2$ with $\|A_1\|, \|A_2\| \leq h$, there cannot be more than $p(|x|) \cdot t(p(|x|))$ strings $\alpha \in U_1$ (or, $\alpha \in U_2$) making $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha\})}(x)$ reject. Hence $\|C_1\| \leq p(|x|) \cdot t(p(|x|))$ and $\|C_2\| \leq p(|x|) \cdot t(p(|x|))$.

For the other case, suppose that $N^{\mathcal{L}(\mathcal{O})}(x)$ rejects. To get a contradiction, assume that $\min\{\|C_1\|, \|C_2\|\} > 2 \cdot p(|x|) \cdot t(p(|x|)) \cdot (p(|x|) \cdot t(p(|x|)) + 1)$. For every $\alpha \in C_1 \cup C_2$, denote by $s(\alpha)$ the unique accepting computation path in $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha\})}(x)$. Define an equivalence relation σ_1 on C_1 as follows: For all $\alpha_1, \alpha_2 \in C_1$,

$$\alpha_1 \sigma_1 \alpha_2 \iff s(\alpha_1) = s(\alpha_2).$$

Let C_1/σ_1 be the quotient set of C_1 determined by σ_1 . We first prove that

$$\|C_1/\sigma_1\| \geq \frac{\|C_1\|}{p(|x|) \cdot t(p(|x|))}. \quad (4.a)$$

To this end, consider any $\alpha \in C_1$. Note that for every query $w \in Q(s(\alpha))$, there cannot be more than $t(|w|)$ different strings $\alpha' \in C_1$ such that

$$w \in \mathcal{L}(\mathcal{O} \cup \{\alpha'\}) \iff w \notin \mathcal{L}(\mathcal{O}).$$

This holds because \mathcal{L} is (h, t) -ambiguous and because $\mathcal{L}(\mathcal{O} \cup A)$ is defined for every $A \subseteq U_1$ with $\|A\| \leq h$. Also since there are at most $p(|x|)$ queries $w \in Q(s(\alpha))$ and each such query is of length $\leq p(|x|)$, there cannot be more than $p(|x|) \cdot t(p(|x|))$ strings $\alpha' \in C_1$ such that for some query $w \in Q(s(\alpha))$, it is the case that $w \in \mathcal{L}(\mathcal{O} \cup \{\alpha'\})$ if and only if $w \notin \mathcal{L}(\mathcal{O})$. In other words, for all but $p(|x|) \cdot t(p(|x|))$ strings $\alpha' \in C_1$, the membership in $\mathcal{L}(\mathcal{O})$ of every query $w \in Q(s(\alpha))$ remains unchanged on inclusion of α' to \mathcal{O} . Since by assumption, $N^{\mathcal{L}(\mathcal{O})}(x)$ rejects, it follows that for no more than $p(|x|) \cdot t(p(|x|))$ strings $\alpha' \in C_1$, (accepting) path $s(\alpha)$ appears in $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha'\})}(x)$. Hence there cannot be more than $p(|x|) \cdot t(p(|x|))$ different strings $\alpha' \in C_1$ such that $s(\alpha') = s(\alpha)$. Thus we have proved Statement (4.a).

Analogously, the same can be proved for C_2 with appropriately defined equivalence class σ_2 .

Define \tilde{C}_1 to be a maximal subset of C_1 such that $s(\alpha_1) \neq s(\alpha_2)$ for every $\alpha_1, \alpha_2 \in \tilde{C}_1$ with $\alpha_1 \neq \alpha_2$. Analogously, define \tilde{C}_2 . Clearly,

$$\|\tilde{C}_1\| \geq \frac{\|C_1\|}{p(|x|) \cdot t(p(|x|))} > 2 \cdot (p(|x|) \cdot t(p(|x|)) + 1), \quad (4.b)$$

and

$$\|\tilde{C}_2\| \geq \frac{\|C_2\|}{p(|x|) \cdot t(p(|x|))} > 2 \cdot (p(|x|) \cdot t(p(|x|)) + 1). \quad (4.c)$$

For every $\alpha_1 \in \tilde{C}_1$, let

$$\text{conflicting}(\alpha_1) = \{\beta_2 \in \tilde{C}_2 \mid s(\alpha_1) \text{ does not appear in } N^{\mathcal{L}(\mathcal{O} \cup \{\alpha_1, \beta_2\})}(x) \text{ or } s(\alpha_1) = s(\beta_2)\},$$

and for every $\alpha_2 \in \tilde{C}_2$, let

$$\text{conflicting}(\alpha_2) = \{\beta_1 \in \tilde{C}_1 \mid s(\alpha_2) \text{ does not appear in } N^{\mathcal{L}(\mathcal{O} \cup \{\alpha_2, \beta_1\})}(x) \text{ or } s(\alpha_2) = s(\beta_1)\}.$$

Fix an arbitrary $\alpha \in \tilde{C}_1$. Since \mathcal{L} is (h, t) -ambiguous and $\mathcal{L}(\mathcal{O} \cup A_1 \cup A_2)$ is defined for every $\|A_1\| \leq 1$ and $\|A_2\| \leq h$, it holds that for every query $w \in Q(s(\alpha))$ in $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha\})}(x)$, there cannot be more than $t(|w|)$ different strings $\beta \in \tilde{C}_2$ such that

$$w \in \mathcal{L}(\mathcal{O} \cup \{\alpha, \beta\}) \iff w \notin \mathcal{L}(\mathcal{O} \cup \{\alpha\}).$$

Also, since no more than $p(|x|)$ strings are queried on each path in $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha\})}(x)$, since each queried string is of length $\leq p(|x|)$, and since for distinct α_1 and α_2 , $s(\alpha_1) \neq s(\alpha_2)$ whenever $\{\alpha_1, \alpha_2\} \subseteq \tilde{C}_2$, we have that for any $\alpha \in \tilde{C}_1$,

$$|\text{conflicting}(\alpha)| \leq p(|x|) \cdot t(p(|x|)) + 1.$$

Analogously, we can prove that for any $\alpha' \in \tilde{C}_2$,

$$|\text{conflicting}(\alpha')| \leq p(|x|) \cdot t(p(|x|)) + 1.$$

With the lower bounds on $|\tilde{C}_1|$ and $|\tilde{C}_2|$ given by Eq. (4.b) and (4.c), it now follows by a simple counting argument that there is a pair $(\alpha_1, \alpha_2) \in \tilde{C}_1 \times \tilde{C}_2$ such that $\alpha_2 \notin \text{conflicting}(\alpha_1)$ and $\alpha_1 \notin \text{conflicting}(\alpha_2)$. Take two such strings α_1 and α_2 . It is easy to see that both $s(\alpha_1)$ and $s(\alpha_2)$ appear in $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha_1, \alpha_2\})}(x)$. Furthermore, $s(\alpha_1) \neq s(\alpha_2)$. Hence, $N^{\mathcal{L}(\mathcal{O} \cup \{\alpha_1, \alpha_2\})}(x)$ has two different accepting computation paths: $s(\alpha_1)$ and $s(\alpha_2)$. This gives a contradiction to the assumption that $N^{\mathcal{L}(\mathcal{O} \cup A_1 \cup A_2)}$ is unambiguous whenever $A_1 \subseteq U_1$ and $A_2 \subseteq U_2$ with $\|A_1\| \leq 1$ and $\|A_2\| \leq 1$. \blacksquare (Lemma 4.8)

Proof of Theorem 4.7. The test language $L(\mathcal{A})$ for our oracle construction is inspired by the one in [CHV93, Theorem 3.1]. For length n , we will reserve the following segment of $k(n)+1$ regions $S_{n,f} = 1^n 0 1^f 0 \Sigma^n$, where $f \in [k(n)+1]$. For $n \geq 1$, define $S_n = \bigcup_{f=1}^{k(n)+1} S_{n,f}$. For all $n \geq 1$ and $f \in [k(n)+1]$, we stipulate that $\|\mathcal{A} \cap S_{n,f}\| \leq h$. Let

$$L(\mathcal{A}) = \{0^n \mid \|\mathcal{A} \cap S_n\| \geq 1\}.$$

Clearly, as long as the oracle set \mathcal{A} maintains the stipulation that $\|\mathcal{A} \cap S_{n,f}\| \leq h$, we have $L(\mathcal{A}) \in \mathbf{R}_{(k(n)+1)\text{-}dt}^p(\mathbf{UP}_{\leq h}^{\mathcal{A}})$ and hence $L(\mathcal{A}) \in \mathbf{R}_{(k(n)+1)\text{-}dt}^p(\mathbf{US}_h^{p,\mathcal{A}})$. We construct an oracle \mathcal{A} such that $L(\mathcal{A}) \notin \mathbf{R}_{k(n)\text{-}T}^p(\mathbf{US}_h^{p,\mathcal{A}})$.³

³The construction can be easily modified to prove the stronger result that $L(\mathcal{A}) \notin \mathbf{R}_{k(n)\text{-}T}^{p,\mathcal{A}}(\mathbf{US}_h^{p,\mathcal{A}})$.

We give a brief informal outline of how the diagonalization for $L(\mathcal{A}) \notin R_{k(n)-T}^p(\text{US}_h^{p,\mathcal{A}})$ is carried out. Fix some input 0^n . The crucial fact is that we have $k(n) + 1$ regions but only $k(n)$ adaptive Turing queries. There are two cases. The easier case is when we can destroy the unambiguity of one of the machines defining the $\text{US}_h^{p,\mathcal{A}}$ set by adding some strings to the current segment (but, of course without violating the above stipulation, i.e., the stipulation that $\|\mathcal{A} \cap S_{n,f}\| \leq h$ for each $f \in [k(n) + 1]$). In that case we can simply add these strings to the oracle and move to the next stage. Otherwise, we can use Lemma 4.8 to show that each Turing query is *insensitive* to all but one of the $k(n) + 1$ regions. A Turing query β is *insensitive* to a region if adding a single string α to that region does not change the answer to β , unless the string α comes from a very small (i.e., polynomially bounded) number of exceptions. But we have only $k(n)$ Turing queries. That's why, there must be a region $S_{n,f}$ to that all Turing queries are insensitive. Since $S_{n,f}$ has exponentially many strings but only polynomially many exceptions, there must be at least one string $\alpha^* \in S_{n,f}$ that we can add to the current segment without changing the answers to any of the Turing queries, and hence without changing the decision (i.e, acceptance or rejection) of the deterministic machine M making the Turing reduction. We add this string α^* to the oracle (thereby changing the membership of 0^n in the test language) if and only if M rejects. This completes the construction in the current stage, and so we move to the next stage.

Now we come to the formal description of the diagonalization. Let $(N_{i,1}, N_{i,2}, \dots, N_{i,h}, M_j)$ be an enumeration of tuples where $N_{i,*}$ is a nondeterministic polynomial-time oracle Turing machine, and M_j is a deterministic polynomial-time oracle Turing machine making, for any set \mathcal{A} and any input of length n , at most $k(n)$ queries to $L[\mathcal{A}; N_{i,1}, N_{i,2}, \dots, N_{i,h}]$ and at most polynomially many queries to \mathcal{A} . Initially, let $\mathcal{A} := \emptyset$.

Stage $\langle i, j \rangle$: Let $p(\cdot)$ be a polynomial that bounds the running time of both $N_{i,*}$ and M_j . Choose an integer n satisfying the following requirements: (a) $2^n > 2 \cdot k(n) \cdot p(p(n)) \cdot t(p(p(n))) \cdot (p(p(n)) \cdot t(p(p(n))) + 1)$, where $t(\cdot)$ is a polynomial defined later in this proof, (b) n is large enough so that n satisfies any promises made in the previous stages and no string of length greater than or equal to n is queried in any of the previous stages, and (c) n is larger than the value of n in the previous stage.

If there exists a set $B \subseteq S_n$ satisfying $\|B \cap S_{n,f}\| \leq h$ for every $f \in [k(n) + 1]$ such that $[\mathcal{A} \cup B; N_{i,1}, N_{i,2}, \dots, N_{i,h}]$ is not unambiguous, then set $\mathcal{A} := \mathcal{A} \cup B$. Promise to choose the value of n in the next stage to be larger than $(p\circ)^h(|w|)$, where w is an arbitrary input string witnessing that $[\mathcal{A} \cup B; N_{i,1}, N_{i,2}, \dots, N_{i,h}]$ is not unambiguous, and then move to the next stage.

Otherwise proceed as follows. Let $\mathcal{L} : \wp(\Sigma^*) \rightarrow \wp(\Sigma^*)$ be the function defined so that for every $\mathcal{O} \subseteq \Sigma^*$,

$$\mathcal{L}(\mathcal{O}) =_{df} L_{\text{unambiguous}}[\mathcal{O}; N_{i,2}, N_{i,3}, \dots, N_{i,h}].$$

By Lemma 3.1, \mathcal{L} is (h, t) -ambiguous for some polynomial $t(\cdot)$. (To be specific, $t(\cdot) = 5^{h-1} \cdot \prod_{i=1}^{h-1} (p\circ)^i(\cdot)$.) Note that $\mathcal{L}(\mathcal{A} \cup B)$ is defined for every $B \subseteq S_n$ satisfying $\|B \cap S_{n,f}\| \leq h$ for every $f \in [k(n) + 1]$.

If $M_j(0^n)$ with oracle $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$ is accepting, then move on to the next stage. If $M_j(0^n)$ with oracle $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$ is rejecting, then look for a string $\alpha \in S_n$ that can be added to \mathcal{A} without changing the decision (i.e., acceptance or rejection) of $M_j(0^n)$ with oracle $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$. Set $\mathcal{A} := \mathcal{A} \cup \{\alpha\}$ and move to the next stage.

End of Stage

It remains to show that such a string α always exists. Consider $M_j(0^n)$ with oracle $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$. Let $\beta_1, \beta_2, \dots, \beta_{k(n)}$ be the sequence of queries made by $M_j(0^n)$ to the oracle $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$. The following claim states that, for any $\beta \in \Sigma^*$, there is one special region $S_{n,\text{sensitive}(\beta)}$ such that, for all regions $S_{n,f}$ different from $S_{n,\text{sensitive}(\beta)}$, and for all but polynomially many $\alpha \in S_{n,f}$, the decision (i.e., acceptance or rejection) of $N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta)$ remains unchanged on addition of α to \mathcal{A} .

Claim 2 *For each $\beta \in \Sigma^*$, there is an integer $\text{sensitive}(\beta) \in [k(n) + 1]$ such that the following is true:*

For every $f \in [k(n) + 1] - \{\text{sensitive}(\beta)\}$, there is a set $C_f(\beta) \subseteq S_{n,f}$ with $|C_f(\beta)| \leq 2 \cdot p(|\beta|) \cdot t(p(|\beta|)) \cdot (p(|\beta|) \cdot t(p(|\beta|)) + 1)$ such that, for every $\alpha \in S_{n,f} - C_f(\beta)$,

$$\beta \in L(N_{i,1}^{\mathcal{L}(\mathcal{A})}) \iff \beta \in L(N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha\})}).$$

Let us assume that the claim is true. There are $k(n) + 1$ regions in S_n , but only $k(n)$ queries $\beta_1, \beta_2, \dots, \beta_{k(n)}$ made by $M_j(0^n)$ to $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$. Let $\ell \in [k(n) + 1]$ such that $\ell \neq \text{sensitive}(\beta_e)$ for every $e \in [k(n)]$. Let $C = C_\ell(\beta_1) \cup C_\ell(\beta_2) \cup \dots \cup C_\ell(\beta_k)$. It is easy to see that we can add any string in $S_{n,\ell} - C$ to \mathcal{A} without changing the decision of $N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta_e)$ for any $e \in [k(n)]$.

Let α be any string in $S_{n,\ell} - C$. Such a string exists because $2 \cdot k(n) \cdot p(|\beta|) \cdot t(p(|\beta|)) \cdot (p(|\beta|) \cdot t(p(|\beta|)) + 1) < 2^n$. Clearly, $M_j(0^n)$ with its $k(n)$ queries to $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$ accepts if and only if $M_j(0^n)$ with its $k(n)$ queries to $L(N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha\})})$ accepts. ■ (Theorem 4.7)

Proof of Claim 2. Suppose the claim were false. Then there exists $\beta \in \Sigma^*$ and $f_1, f_2 \in [k(n) + 1]$ with $f_1 \neq f_2$ such that

- $C_{f_1}(\beta) =_{df} \{ \alpha \in S_{n,f_1} \mid \beta \in L(N_{i,1}^{\mathcal{L}(\mathcal{A})}) \iff \beta \notin L(N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha\})}) \}$ with $|C_{f_1}(\beta)| > 2 \cdot p(|\beta|) \cdot t(p(|\beta|)) \cdot (p(|\beta|) \cdot t(p(|\beta|)) + 1)$, and
- $C_{f_2}(\beta) =_{df} \{ \alpha \in S_{n,f_2} \mid \beta \in L(N_{i,1}^{\mathcal{L}(\mathcal{A})}) \iff \beta \notin L(N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha\})}) \}$ with $|C_{f_2}(\beta)| > 2 \cdot p(|\beta|) \cdot t(p(|\beta|)) \cdot (p(|\beta|) \cdot t(p(|\beta|)) + 1)$.

Apply Lemma 4.8 with $\mathcal{O} := \mathcal{A}$, $U_1 := S_{n,f_1}$, and $U_2 := S_{n,f_2}$. We obtain $\min\{|C_{f_1}(\beta)|, |C_{f_2}(\beta)|\} \leq 2 \cdot p(|\beta|) \cdot t(p(|\beta|)) \cdot (p(|\beta|) \cdot t(p(|\beta|)) + 1)$, a contradiction. ■ (Claim 2)

4.3 Simulating Nonadaptive Access by Adaptive Access (Promise Case)

Cai, Hemachandra, and Vyskoč [CHV93] proved the following partial improvement of their Theorem 4.6.

Theorem 4.9 ([CHV93]) *For any constant k , there exists an oracle \mathcal{A} such that*

$$R_{(k+1)-tt}^p(\text{UP}^{\mathcal{A}}) \not\subseteq R_{s,k-T}^{p,\mathcal{A}}(\text{Promise-UP}^{\mathcal{A}}).$$

Note that we have replaced “UP” by “Promise-UP” on the righthand side of the noninclusion relation of Theorem 4.6. This is a significant improvement for the following reason. The computational powers of $R_b^p(\text{UP})$ and $R_{s,b}^p(\text{Promise-UP})$ (the bounded Turing closure of UP and the bounded smart Turing closure of Promise-UP, respectively) are known to be remarkably different in certain relativized worlds. While it is easy to show that $\text{UP}_{\leq k}$ is robustly (i.e., for every oracle) contained in $\text{P}_{s,k-T}^{\text{Promise-UP}}$ for any $k \geq 1$, we have shown in the proof of Theorem 4.4 that for no $k \geq 2$, $\text{UP}_{\leq k}$ is robustly contained in P^{UP} . Therefore, it is not immediately clear whether this improvement is impossible, i.e., whether $R_{(k+1)-tt}^p(\text{UP}) \subseteq R_{s,k-T}^p(\text{Promise-UP})$ holds relative to all oracles.

However, Cai, Hemachandra, and Vyskoč [CHV93, Theorem 3.1] could achieve this improvement only by paying a heavy price. In their own words:

In our earlier version dealing with $\text{UP}^{\mathcal{A}}$, the constant k can be replaced by any arbitrary polynomial-time computable function $f(n)$ with polynomially bounded value. It remains open whether the claim of the current strong version of Theorem 3.1 can be similarly generalized to non-constant access.

We resolve this open question. We show that Theorem 4.9 holds with constant k replaced by any total, polynomial-time computable and polynomially bounded function $k(\cdot)$. This result is subsumed as the special case $h = 1$ of our main result, Theorem 4.10, of this subsection.

Theorem 4.10 *For any total, polynomial-time computable and polynomially bounded function $k(\cdot)$, and integer $h \geq 1$, there exists an oracle \mathcal{A} such that*

$$R_{(k(n)+1)-dtt}^p(\text{UP}_{\leq h}^{\mathcal{A}}) \not\subseteq R_{s,k(n)-T}^{p,\mathcal{A}}(\text{Promise-UP}^{\text{U}\Sigma_{h-1}^{p,\mathcal{A}}}),$$

and hence $R_{(k(n)+1)-dtt}^p(\text{U}\Sigma_h^{p,\mathcal{A}}) \not\subseteq R_{s,k(n)-T}^{p,\mathcal{A}}(\text{Promise-UP}^{\text{U}\Sigma_{h-1}^{p,\mathcal{A}}})$.

Theorem 4.10 is furthermore a generalization of Theorem 4.9 to higher levels of the unambiguous polynomial hierarchy.

The proof of Theorem 4.10 is much more challenging than the proof of Theorem 4.7 because we now require diagonalizing against $R_{s,k(n)-T}^{p,\mathcal{A}}(\text{Promise-UP}^{\text{U}\Sigma_{h-1}^{p,\mathcal{A}}})$ as opposed to diagonalizing against $R_{k(n)-T}^{p,\mathcal{A}}(\text{UP}^{\text{U}\Sigma_{h-1}^{p,\mathcal{A}}})$. To diagonalize against $R_{k(n)-T}^{p,\mathcal{A}}(\text{UP}^{\text{U}\Sigma_{h-1}^{p,\mathcal{A}}})$ as in the proof of Theorem 4.7, it was sufficient at any stage to extend the current oracle \mathcal{A} so that the $\Sigma_h(\mathcal{A})$ -system (corresponding to the stage) becomes ambiguous on some input string, even if the input string witnessing the ambiguity of the $\Sigma_h(\mathcal{A})$ -system would never arise in a valid computation of the deterministic querying machine (corresponding to the same stage). This is, however, not sufficient when we diagonalize against $R_{s,k(n)-T}^{p,\mathcal{A}}(\text{Promise-UP}^{\text{U}\Sigma_{h-1}^{p,\mathcal{A}}})$.

Any input string witnessing the ambiguity of the $\Sigma_h(\mathcal{A})$ -system must now have its origin from a valid computation of the deterministic querying machine.

To prove Theorem 4.9, Cai et al. [CHV93] presented the following combinatorial lemma.

Lemma 4.11 (The Gaming Lemma [CHV93]) *For $1 \leq i \leq m$, let \mathcal{S}_i be a collection of nonempty subsets of $[n]$ with the following properties:*

1. $(\forall j \in [n])(\exists \ell \in [m])(\{j\} \in \mathcal{S}_\ell)$, and
2. $(\forall j \in [m])[(A, B \in \mathcal{S}_j \text{ and } A \neq B) \implies (\exists \ell \in [j-1])(\exists C \in \mathcal{S}_\ell)[C \subseteq A \cup B]]$.

Then $m \geq n$.

Cai, Hemachandra, and Vyskoč [CHV93] gave the following informal interpretation of this lemma. Suppose a combinatorial game is to be played given the set $[n]$ at hand. The game is played in steps by a single player. At each step i , the player can generate a collection \mathcal{S}_i of nonempty subsets of $[n]$ with a restriction: If sets $A, B \subseteq [n]$ are generated at some step $i > 1$, then there must be a previous step $j < i$ and a set C generated at that step such that $C \subseteq A \cup B$. The game ends as soon as all the singletons $\{k\} \subseteq [n]$ are eventually produced. The gaming lemma states that this combinatorial game requires at least n steps.

Our proof of Theorem 4.10 also makes use of the gaming lemma. However, the actual diagonalization steps are considerably different from the one in [CHV93]. The most tricky part is the proof of Lemma 4.12. It demonstrates the existence of especially *nice* strings $\alpha_1, \alpha_2, \dots, \alpha_r$ satisfying certain useful properties. Each string α_i serves as a representative for one region $S_{n,i}$ of the oracle. These strings satisfy a kind of *independence* property in the following sense. Let A and B be two different minimal subsets of $\{\alpha_1, \alpha_2, \dots, \alpha_r\}$ that are minimal in the sense that adding A or B to an oracle makes an oracle NPTM N to accept, but adding any proper subset of A or B to the oracle makes N to reject. Then the independence property implies that adding all the strings in $A \cup B$ to the oracle will make N to have at least two accepting paths.

In each stage of the diagonalization, there are two cases. The easier case is when we can destroy the unambiguity of the $\text{US}_{h-1}^{p,\mathcal{A}}$ oracle in the $\text{NP}^{\text{US}_{h-1}^{p,\mathcal{A}}}$ -machine, the machine against that we are diagonalizing, by adding some strings (but, of course without violating certain requirements of the stage) to the oracle \mathcal{A} . In that case, we can simply add these strings to the oracle \mathcal{A} and move to the next stage. Otherwise, we apply the gaming lemma (Lemma 4.11) to show that by adding only the *nice* strings α_i to the oracle \mathcal{A} , the desired diagonalization step for the current stage can be achieved. The existence of these *nice* strings α_i was a key idea that led us to the resolution of the question by Cai, Hemachandra, and Vyskoč [CHV93] and in generalizing their result.

Lemma 4.12 *Fix an oracle NPTM N with running time bounded by some polynomial $p(\cdot)$, a set $\mathcal{O} \subseteq \Sigma^*$, and sets $X = \{x_1, x_2, \dots, x_d\} \subseteq (\Sigma^*)^{\leq m}$ and $Y = \{y_1, y_2, \dots, y_d\} \subseteq (\Sigma^*)^{\leq m}$. Let $U_1, U_2, \dots, U_r \subseteq \Sigma^*$ be such that for each distinct $U_\ell, U_{\ell'}$, it holds that $\mathcal{O} \cap U_\ell = \mathcal{O} \cap U_{\ell'} = U_\ell \cap U_{\ell'} = \emptyset$ and $\|U_\ell\| = \|U_{\ell'}\| \geq u$. Let \mathcal{L} be an (h, t) -ambiguous function such that $\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in [r]} A_\ell))$ is defined for every $A_\ell \subseteq U_\ell$ satisfying $\|A_\ell\| \leq h$. If*

$u > (3 \cdot d + d') \cdot r \cdot 2^{2r} \cdot p(m) \cdot t(p(m))$, then there exist strings $\alpha_1 \in U_1, \alpha_2 \in U_2, \dots, \alpha_r \in U_r$ such that the following properties hold:

(A) For every $x \in X$ and for every pair of distinct, nonempty sets $S_1, S_2 \in \wp(\{1, 2, \dots, r\})$, if the conditions:

(A.1) $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1} \{\alpha_\ell\}))}(x)$ accepts and for all $S'_1 \subset S_1$, $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S'_1} \{\alpha_\ell\}))}(x)$ rejects, and

(A.2) $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_2} \{\alpha_\ell\}))}(x)$ accepts and for all $S'_2 \subset S_2$, $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S'_2} \{\alpha_\ell\}))}(x)$ rejects,

are satisfied, then there are at least two accepting paths in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cup S_2} \{\alpha_\ell\}))}(x)$.

(B) For every $y \in Y$ and for every nonempty set $S \in \wp(\{1, 2, \dots, r\})$, the following is true:

if $N^{\mathcal{L}(\mathcal{O})}(y)$ accepts, then $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\}))}(y)$ also accepts.

Proof We prove the lemma using the probabilistic method. Choose $\alpha_1 \in U_1, \alpha_2 \in U_2, \dots, \alpha_r \in U_r$ uniformly and independently at random. Let E_1 be the event that there exist an $x \in X$ and distinct, nonempty sets $S_1, S_2 \in \wp(\{1, 2, \dots, r\})$ satisfying the conditions (A.1) and (A.2) given in the lemma, but $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cup S_2} \{\alpha_\ell\}))}(x)$ has at most one accepting path. Let E_2 be the event that there exist a $y \in Y$ and a nonempty set $S \in \wp(\{1, 2, \dots, r\})$ such that $N^{\mathcal{L}(\mathcal{O})}(y)$ accepts, but $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\}))}(y)$ rejects. We will prove that $\text{Prob}(E_1) + \text{Prob}(E_2) < 1$, thus completing the proof of the lemma.

We first make the following claim.

Claim 3 Fix a string $z \in \Sigma^*$. Let $\mathcal{O}, U_1, U_2, \dots, U_r$ be sets defined in the statement of Lemma 4.12. Let \mathcal{L} be an (h, t) -ambiguous function such that $\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in [r]} A_\ell))$ is defined for every $A_\ell \subseteq U_\ell$ satisfying $|A_\ell| \leq h$. Let $T \subseteq [r]$. Fix $\alpha_i \in U_i$ for each $i \in T$ arbitrarily. If we choose $\alpha_j \in U_j$, for each $j \in [r] - T$, uniformly and independently at random, then for any $T_1, T_2 \subseteq [r]$ satisfying $(T_1 \Delta T_2) \cap T = \emptyset$,

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_1} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_2} \{\alpha_\ell\}))$$

is true with probability $\leq r \cdot t(|z|)/u$.

Proof of Claim 3. Let $V = T_1 \cap T_2$. Because \mathcal{L} is (h, t) -ambiguous, for any $j \in T_1 \Delta T_2$, the probability over uniform random choice of $\alpha_j \in U_j$ that

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in V} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in V} \{\alpha_\ell\}) \cup \{\alpha_j\})$$

is at most $t(|z|)/u$. Successively choose $\alpha_j \in U_j$ uniformly at random, where $j \in T_1 \Delta T_2$, and add j to V until V equals T_1 . The probability that

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_1} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_1 \cap T_2} \{\alpha_\ell\}))$$

is, therefore, at most $\|T_1 - T_2\| \cdot t(|z|)/u$. Likewise, the probability that

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_2} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_1 \cap T_2} \{\alpha_\ell\}))$$

is at most $\|T_2 - T_1\| \cdot t(|z|)/u$. Hence the probability that

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_1} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in T_2} \{\alpha_\ell\}))$$

is at most $(\|T_1 - T_2\| + \|T_2 - T_1\|) \cdot t(|z|)/u \leq r \cdot t(|z|)/u$. ■ (Claim 3)

Using Claim 3, we obtain an upper bound on $\text{Prob}(E_1)$ in Claim 4, and an upper bound on $\text{Prob}(E_2)$ in Claim 5.

Claim 4 $\text{Prob}(E_1) \leq 3 \cdot d \cdot r \cdot 2^{2r} \cdot p(m) \cdot t(p(m))/u$.

Proof of Claim 4. Let C_{x,S_1,S_2} stand for the condition “input x and the pair S_1, S_2 satisfy the conditions (A.1) and (A.2) given in the lemma.” Fix an $x \in X$ and distinct, nonempty sets S_1 and S_2 . Let E_{x,S_1,S_2} denote the event that C_{x,S_1,S_2} is satisfied, but $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cup S_2} \{\alpha_\ell\}))}(x)$ has at most one accepting path. Clearly, $\text{Prob}(E_1) \leq \sum_{x,S_1,S_2} \text{Prob}(E_{x,S_1,S_2})$. If condition C_{x,S_1,S_2} is satisfied, then for each $i \in \{1, 2\}$, we can fix an accepting path $\rho(S_i)$ in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_i} \{\alpha_\ell\}))}(x)$. For definiteness, let $\rho(S_i)$ be the lexicographically first accepting path in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_i} \{\alpha_\ell\}))}(x)$.

If E_{x,S_1,S_2} occurs, then at least one of the events J , $H(S_1)$, or $H(S_2)$ occurs, where

- J is the event that condition C_{x,S_1,S_2} is satisfied, and $\rho(S_1)$ and $\rho(S_2)$ are equal.
- $H(S_1)$ is the event that condition C_{x,S_1,S_2} is satisfied and $\rho(S_1)$ does not appear in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cup S_2} \{\alpha_\ell\}))}(x)$.
- $H(S_2)$ is the event that condition C_{x,S_1,S_2} is satisfied and $\rho(S_2)$ does not appear in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cup S_2} \{\alpha_\ell\}))}(x)$.

We first determine an upper bound on $\text{Prob}(J)$. To this end, we determine an upper bound on $\text{Prob}(J \mid \alpha_i = \beta_i \text{ for all } i \in S_1)$ for arbitrary fixed strings $\beta_i \in U_i$ for each $i \in S_1$. Hence, suppose henceforth that $\alpha_i = \beta_i$ for all $i \in S_1$. Because $S_1 \neq S_2$ and because we can assume that S_1 satisfies the condition (A.1) given in the lemma, the (accepting) path $\rho(S_1)$ in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1} \{\alpha_\ell\}))}(x)$ does not appear in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cap S_2} \{\alpha_\ell\}))}(x)$. Hence for at least one string $z \in Q(\rho(S_1))$ queried along $\rho(S_1)$, we have

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cap S_2} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1} \{\alpha_\ell\})). \quad (4.d)$$

Note that this condition depends only on strings α_i with $i \in S_1$, which are fixed by $\alpha_i = \beta_i$. Fix one string z satisfying Statement (4.d). Applying Claim 3 with $T := S_1$, $T_1 := S_1 \cap S_2$,

and $T_2 := S_2$, we get that

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cap S_2} \{\alpha_\ell\})) \iff z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_2} \{\alpha_\ell\})) \quad (4.e)$$

holds with probability $\geq 1 - r \cdot t(|z|)/u \geq 1 - r \cdot t(p(m))/u$. Statements (4.d) and (4.e) together imply that with probability $\geq 1 - r \cdot t(p(m))/u$,

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_2} \{\alpha_\ell\})). \quad (4.f)$$

Hence with probability $\geq 1 - r \cdot t(p(m))/u$, path $\rho(S_1)$ does not appear in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_2} \{\alpha_\ell\}))}(x)$, and therefore $\rho(S_1) \neq \rho(S_2)$. We have proven that for arbitrary fixed strings $\{\beta_i \in U_i \mid i \in S_1\}$, it holds that $\text{Prob}(J \mid \alpha_i = \beta_i \text{ for all } i \in S_1) \leq r \cdot t(p(m))/u$. Therefore, also $\text{Prob}(J) \leq r \cdot t(p(m))/u$.

To determine an upper bound on $\text{Prob}(H(S_1))$, we determine an upper bound on $\text{Prob}(H(S_1) \mid \alpha_i = \beta_i \text{ for all } i \in S_1)$ for arbitrary fixed strings $\beta_i \in U_i$ for each $i \in S_1$. Hence, suppose henceforth that $\alpha_i = \beta_i$ for all $i \in S_1$. Clearly, $\rho(S_1)$ depends only on strings α_i with $i \in S_1$, which we have fixed by $\alpha_i = \beta_i$. Then the event $H(S_1)$ occurs only if it holds that $\rho(S_1)$ queries some string z with

$$z \in \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1} \{\alpha_\ell\})) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1 \cup S_2} \{\alpha_\ell\})).$$

Note that path $\rho(S_1)$ in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S_1} \{\alpha_\ell\}))}(x)$ queries at most $p(m)$ strings, each of which is of length $\leq p(m)$. Applying Claim 3 with $T := S_1$, $T_1 := S_1$, and $T_2 := S_1 \cup S_2$, we get that $\text{Prob}(H(S_1) \mid \alpha_i = \beta_i \text{ for all } i \in S_1) \leq p(m) \cdot r \cdot t(p(m))/u$. We have thus proven that for arbitrary fixed strings $\{\beta_i \in U_i \mid i \in S_1\}$, it holds that $\text{Prob}(H(S_1) \mid \alpha_i = \beta_i \text{ for all } i \in S_1) \leq p(m) \cdot r \cdot t(p(m))/u$. Therefore, also $\text{Prob}(H(S_1)) \leq p(m) \cdot r \cdot t(p(m))/u$.

Analogously, we can prove that $\text{Prob}(H(S_2)) \leq p(m) \cdot r \cdot t(p(m))/u$.

Thus $\text{Prob}(E_{x, S_1, S_2}) \leq \text{Prob}(J \cup H(S_1) \cup H(S_2)) \leq \text{Prob}(J) + \text{Prob}(H(S_1)) + \text{Prob}(H(S_2)) \leq 3 \cdot p(m) \cdot r \cdot t(p(m))/u$. It follows that $\text{Prob}(E_1) \leq 3 \cdot d \cdot r \cdot 2^{2r} \cdot p(m) \cdot t(p(m))/u$. ■ (Claim 4)

Claim 5 $\text{Prob}(E_2) \leq d' \cdot r \cdot 2^r \cdot p(m) \cdot t(p(m))/u$.

Proof of Claim 5. Fix a $y \in Y$ and a nonempty set $S \in \wp(\{1, 2, \dots, r\})$. Let $E_{y, S}$ denote the event that $N^{\mathcal{L}(\mathcal{O})}(y)$ accepts, but $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\}))}(y)$ rejects. Clearly, $\text{Prob}(E_2) \leq \sum_{y, S} \text{Prob}(E_{y, S})$. If $N^{\mathcal{L}(\mathcal{O})}(y)$ accepts, then we can fix an accepting path ρ in $N^{\mathcal{L}(\mathcal{O})}(y)$. For definiteness, let ρ be the lexicographically first accepting path in $N^{\mathcal{L}(\mathcal{O})}(y)$.

Suppose $E_{y, S}$ occurs. Since $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\}))}(y)$ rejects, the (accepting) path ρ in $N^{\mathcal{L}(\mathcal{O})}(y)$ does not appear in $N^{\mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\}))}(y)$. Hence for at least one string $z \in Q(\rho)$ queried along ρ , we have

$$z \in \mathcal{L}(\mathcal{O}) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\})).$$

Applying Claim 3 with $T := \emptyset$, $T_1 := \emptyset$, and $T_2 := S$, we get that for each $z \in Q(\rho)$,

$$z \in \mathcal{L}(\mathcal{O}) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\}))$$

holds with probability $\leq r \cdot t(|z|)/u$. Since $\|Q(\rho)\| \leq p(m)$ and since the length of each query $z \in Q(\rho)$ is at most $p(m)$, with probability $\leq p(m) \cdot r \cdot t(p(m))/u$ we have, for some $z \in Q(\rho)$,

$$z \in \mathcal{L}(\mathcal{O}) \iff z \notin \mathcal{L}(\mathcal{O} \cup (\bigcup_{\ell \in S} \{\alpha_\ell\})).$$

Thus we have shown that $\text{Prob}(E_{y,S}) \leq p(m) \cdot r \cdot t(p(m))/u$. It follows that $\text{Prob}(E_2) \leq d' \cdot r \cdot 2^r \cdot p(m) \cdot t(p(m))/u$. \blacksquare (Claim 5)

From Claim 4 and Claim 5, we get $\text{Prob}(E_1) + \text{Prob}(E_2) \leq (3 \cdot d + d') \cdot r \cdot 2^{2r} \cdot p(m) \cdot t(p(m))/u < 1$, by our choice of u . This completes the proof. \blacksquare (Lemma 4.12)

Now it is relatively easy to prove the main result of this subsection.

Proof of Theorem 4.10. For each length n , we will reserve the following segment of $k(n) + 1$ regions $S_{n,f} = 1^n 0 1^f 0 \Sigma^{3k(n)+n}$, where $f \in [k(n) + 1]$. For $n \geq 1$, define $S_n = \bigcup_{f=1}^{k(n)+1} S_{n,f}$. We take the test language $L(\mathcal{A})$ used in the proof of Theorem 4.7. The oracle \mathcal{A} is constructed in stages. Let $(N_{i,1}, N_{i,2}, \dots, N_{i,h}, M_j)$ be an enumeration of tuples where $N_{i,*}$ is a nondeterministic polynomial-time oracle Turing machine, and M_j is a deterministic polynomial-time oracle Turing machine making, for any set \mathcal{A} and any input of length n , at most $k(n)$ queries to $L[\mathcal{A}; N_{i,1}, N_{i,2}, \dots, N_{i,h}]$ and at most polynomially many queries to \mathcal{A} . Initially $\mathcal{A} := \emptyset$.

Stage $\langle i, j \rangle$: Let $p(\cdot)$ be a polynomial that bounds the running time of both $N_{i,*}$ and M_j . Choose a very large integer n such that (a) $2^{3k(n)+n} - p(n) > 4 \cdot k(n) \cdot (k(n) + 1) \cdot 2^{2(k(n)+1)} \cdot p(p(n)) \cdot t(p(p(n)))$, where $t(\cdot)$ is a polynomial defined later in this proof, (b) no string of length n or more is queried in any of the previous stages, (c) n is larger than the value in the previous stage, and (d) n satisfies any promises made in the previous stages.

If there exists a set $B \subseteq S_n$, satisfying $\|B \cap S_{n,f}\| \leq h$ for every $f \in [k(n) + 1]$, such that (a) $[\mathcal{A} \cup B; N_{i,2}, N_{i,3}, \dots, N_{i,h}]$ is not unambiguous, or (b) $M_j(0^n)$ queries β to $[\mathcal{A} \cup B; N_{i,1}, N_{i,2}, \dots, N_{i,h}]$ and $N_{i,1}^{L[\mathcal{A} \cup B; N_{i,2}, N_{i,3}, \dots, N_{i,h}]}(\beta)$ is not unambiguous, then set $\mathcal{A} := \mathcal{A} \cup B$. Promise to choose the value of n in the next stage to be sufficiently large so that any of the requirements (a), (b) satisfied in this stage cannot become invalid in the next stage.

Otherwise if no such set $B \subseteq S_n$ exists, then we proceed as follows. Define an (h, t) -ambiguous function $\mathcal{L} : \wp(\Sigma^*) \rightarrow \wp(\Sigma^*)$ so that for every $\mathcal{O} \subseteq \Sigma^*$,

$$\mathcal{L}(\mathcal{O}) =_{df} L_{\text{unambiguous}}[\mathcal{O}; N_{i,2}, N_{i,3}, \dots, N_{i,h}],$$

where $t(\cdot)$ is some polynomial.⁴ It is easy to see that $\mathcal{L}(\mathcal{A} \cup B)$ is defined for every $B \subseteq S_n$, which satisfies $\|B \cap S_{n,f}\| \leq h$ for every $f \in [k(n) + 1]$. We next use Claim 6 to successfully finish this stage.

Claim 6 *There is a string $\alpha \in S_n$ such that $M_j(0^n)$ with oracle $\mathcal{A} \oplus L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$ is identical to $M_j(0^n)$ with oracle $(\mathcal{A} \cup \{\alpha\}) \oplus L(N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha\})})$.*

That is, if $M_j(0^n)$ with oracle $\mathcal{A} \oplus L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$ rejects, then we set $\mathcal{A} := \mathcal{A} \cup \{\alpha\}$; otherwise if $M_j(0^n)$ with oracle $\mathcal{A} \oplus L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$ accepts, then we leave \mathcal{A} unchanged. Finally, we move to the next stage.

End of Stage

This completes the proof of Theorem 4.10. ■ (Theorem 4.10)

Proof of Claim 6. Let $\beta_1, \beta_2, \dots, \beta_{k(n)}$ be the sequence of queries made by $M_j(0^n)$ to the oracle $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$. Let $I = \{\ell \mid N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta_\ell) \text{ accepts}\}$. Let \tilde{Q} be the set of strings that are queried by $M_j(0^n)$ to oracle \mathcal{A} . Clearly, $\|\tilde{Q}\| \leq p(n)$.

Apply Lemma 4.12 with $N := N_{i,1}$, $\mathcal{O} := \mathcal{A}$, $d := k(n)$, $d' := k(n)$, $X = \{\beta_1, \beta_2, \dots, \beta_{k(n)}\}$, $Y = \{\beta_1, \beta_2, \dots, \beta_{k(n)}\}$, $m := p(n)$, $r := k(n) + 1$, $U_f := S_{n,f} - \tilde{Q}$ for each $f \in [k(n) + 1]$, and $u := 2^{3k(n)+n} - p(n)$. We obtain strings $\alpha_1 \in S_{n,1} - \tilde{Q}$, $\alpha_2 \in S_{n,2} - \tilde{Q}$, \dots , $\alpha_{k(n)+1} \in S_{n,k(n)+1} - \tilde{Q}$, which satisfy the properties (A) and (B) given in Lemma 4.12. Now assign to each query β_ℓ with $\ell \in [k(n)] - I$ a collection $\mathcal{S}_\ell \subseteq \mathcal{P}([k(n) + 1])$ in the following way: $\{f_1, f_2, \dots, f_s\} \in \mathcal{S}_\ell$ if and only if

- (a) adding $\{\alpha_{f_1}, \alpha_{f_2}, \dots, \alpha_{f_s}\}$ to \mathcal{A} makes $N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta_\ell)$ change from rejection to acceptance, i.e., $N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta_\ell)$ rejects but $N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha_{f_1}, \alpha_{f_2}, \dots, \alpha_{f_s}\})}(\beta_\ell)$ accepts, and
- (b) no set $T \subset \{f_1, f_2, \dots, f_s\}$ satisfies (a), i.e., for no such set T it holds that $N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha_j \mid j \in T\})}(\beta_\ell)$ accepts.

Note that no collection \mathcal{S}_ℓ contains the empty set. However, some of these collections may be empty.

Suppose that Claim 6 is not true. Then for every $e \in [k(n) + 1]$, there is an $\ell \in [k(n)]$ such that

$$N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta_\ell) \text{ rejects} \iff N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha_e\})}(\beta_\ell) \text{ accepts.} \quad (4.g)$$

Because $\alpha_1, \alpha_2, \dots, \alpha_{k(n)+1}$ satisfy property (B) of Lemma 4.12, Statement (4.g) can only be true for $\ell \notin I$. This implies that for every $e \in [k(n) + 1]$, there is an $\ell \in [k(n)] - I$ such that

$$N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta_\ell) \text{ rejects and } N_{i,1}^{\mathcal{L}(\mathcal{A} \cup \{\alpha_e\})}(\beta_\ell) \text{ accepts.}$$

It follows from the definition of the collections \mathcal{S}_ℓ that for every $e \in [k(n) + 1]$, there is a collection \mathcal{S}_ℓ such that the singleton $\{\alpha_e\}$ is contained in \mathcal{S}_ℓ . Thus we have proven condition (1) of Lemma 4.11 (the gaming lemma).

⁴As observed in the proof of Theorem 4.7, we can take $t(\cdot)$ to be the polynomial $5^{h-1} \cdot \prod_{i=1}^{h-1} (p\alpha)^i(\cdot)$.

Now take two distinct sets $A, B \in \mathcal{S}_\ell$ for some $\ell \in [k(n)]$. Then by the definition of the collections \mathcal{S}_ℓ together with property (A) of Lemma 4.12, $N_{i,1}^{\mathcal{L}(\mathcal{A} \cup (\bigcup_{e \in A \cup B} \{\alpha_e\}))}(\beta_\ell)$ has at least two accepting paths. Because of our assumption about the unambiguity, we can be sure that adding $\bigcup_{e \in A \cup B} \{\alpha_e\}$ to \mathcal{A} changes the decision (i.e., acceptance or rejection) of $N_{i,1}^{\mathcal{L}(\mathcal{A})}$ for a previous query $\beta_{\ell'}$ with $\ell' < \ell$. The decision of $N_{i,1}^{\mathcal{L}(\mathcal{A})}(\beta_{\ell'})$ on addition of $\bigcup_{e \in A \cup B} \{\alpha_e\}$ to \mathcal{A} must change from rejection to acceptance, and not from acceptance to rejection, because $\alpha_1, \alpha_2, \dots, \alpha_{k(n)+1}$ satisfy property (B) of Lemma 4.12. Hence there is a set $C \in \mathcal{S}_{\ell'}$ such that $C \subseteq A \cup B$. This proves condition (2) of Lemma 4.11. Lemma 4.11 implies that the number of queries $k(n)$ is greater than or equal to the number of regions $k(n) + 1$, a contradiction. \blacksquare (Claim 6)

4.4 Simulating Adaptive Access by Nonadaptive Access

Sections 4.2 and 4.3 studied the limitations of simulating nonadaptive queries to $\text{UP}_{\leq h}$ by adaptive queries to US_h^p in relativized settings. This section complements these investigations. In particular, Corollary 4.14 of this section shows that in a certain relativized world, it is impossible to simulate adaptive k -Turing access to $\text{UP}_{\leq h}$ by nonadaptive $(2^k - 2)$ -tt access to US_h^p . This also implies optimality of robustly (i.e., for every oracle) simulating adaptive k -Turing accesses by nonadaptive $(2^k - 1)$ -tt accesses to classes such as $\text{UP}_{\leq h}$ and US_h^p , since for any class \mathcal{C} , we can easily, via a brute-force method, simulate adaptive k -Turing reduction to \mathcal{C} by nonadaptive $(2^k - 1)$ -tt reduction to \mathcal{C} .

The proof of Theorem 4.13 employs a technique of Buhrman, Spaan, and Torenvliet [BST93], which Cai, Hemachandra, and Vyskoč [CHV93] referred to by “force your way through the tree” technique. Buhrman, Spaan, and Torenvliet [BST93] used their technique to prove that NEXP has a set that is complete for k -Turing reductions, but not complete for $(2^k - 2)$ -tt reductions. Cai, Hemachandra, and Vyskoč [CHV93] used this technique to prove Theorem 4.13 for the case of $h = 1$. We use the same approach to generalize the result of Cai, Hemachandra, and Vyskoč [CHV93] from the case of $h = 1$ to the case of arbitrary integer $h \geq 1$.

Theorem 4.13 *For any integers $k, h \geq 1$, there exists an oracle \mathcal{A} such that*

$$R_{k-T}^p(\text{UP}_{\leq h}^{\mathcal{A}}) \not\subseteq R_{(2^k-2)\text{-tt}}^{p,\mathcal{A}}(\text{NP}^{\text{US}_{h-1}^{p,\mathcal{A}}}).$$

Proof For each length n , we will reserve the following segment of $2^k - 1$ regions: $S_{n,f} = 1^n 0 f 0 \Sigma^n$, where $f \in (\Sigma^*)^{\leq k-1}$. For $n \geq 1$, define $S_n = \bigcup_{f \in (\Sigma^*)^{\leq k-1}} S_{n,f}$. For each length n and set $A \subseteq \Sigma^*$, we also define a sequence $b_{n,1}^A b_{n,2}^A \dots b_{n,k}^A$ of bits as follows:

(\star)

$$b_{n,1}^A = \begin{cases} 1 & \text{if } S_{n,\epsilon} \cap A \neq \emptyset, \\ 0 & \text{otherwise,} \end{cases}$$

and for each ℓ with $2 \leq \ell \leq k$,

$$b_{n,\ell}^A = \begin{cases} 1 & \text{if } S_{n,b_{n,1}^A b_{n,2}^A \dots b_{n,\ell-1}^A} \cap A \neq \emptyset, \\ 0 & \text{otherwise.} \end{cases}$$

Our test language is $L(\mathcal{A}) = \{0^n \mid b_{n,k}^A = 1\}$. Note that our test language is the same as the one in [CHV93, Theorem 3.3]. We stipulate that for all $n \geq 1$ and for all $f \in (\Sigma^*)^{\leq k-1}$, $\|S_{n,f}\| \leq h$. Clearly, if the oracle set \mathcal{A} maintains this stipulation, then we have $L(\mathcal{A}) \in R_{k-T}^p(\text{UP}_{\leq h}^A)$. We construct an oracle \mathcal{A} such that $L(\mathcal{A}) \notin R_{(2^k-2)-tt}^p(\text{NP}^{\text{UP}_{h-1}^{p,\mathcal{A}}})$. The construction can be easily modified to prove the stronger result that $L(\mathcal{A}) \notin R_{(2^k-2)-tt}^{p,\mathcal{A}}(\text{NP}^{\text{UP}_{h-1}^{p,\mathcal{A}}})$.

Let $(N_{i,1}, N_{i,2}, \dots, N_{i,h}, M_j)$ be an enumeration of tuples, where $N_{i,*}$ is a nondeterministic polynomial-time oracle Turing machine, and M_j is a deterministic polynomial-time oracle Turing machine making, for any set \mathcal{A} and for any input, at most $2^k - 2$ nonadaptive queries to $L[\mathcal{A}; N_{i,1}, N_{i,2}, \dots, N_{i,h}]$. Initially, let $\mathcal{A} := \emptyset$.

Stage $\langle i, j \rangle$: Let $p(\cdot)$ be a polynomial that bounds the running time of both $N_{i,*}$ and M_j . Choose a very large integer n such that (a) $2^n > 2^{2k} \cdot p(p(n)) \cdot t(p(p(n)))$, where $t(\cdot)$ is a polynomial defined later in this proof, (b) no string of length n or more is queried in any of the previous stages, (c) n is larger than the value in the previous stage, and (d) n satisfies any promises made in the previous stages.

If there exists a set $B \subseteq S_n$ satisfying $\|B \cap S_{n,f}\| \leq h$ for every $f \in (\Sigma^*)^{\leq k-1}$ such that $[\mathcal{A} \cup B; N_{i,2}, \dots, N_{i,h}]$ is not unambiguous, then set $\mathcal{A} := \mathcal{A} \cup B$. Promise to choose the value of n in the next stage to be larger than $(p \circ)^{h-1}(|w|)$, where w is an arbitrary string witnessing that $[\mathcal{A} \cup B; N_{i,2}, \dots, N_{i,h}]$ is not unambiguous, and then move to the next stage.

Otherwise, we define a function $\mathcal{L} : \wp(\Sigma^*) \rightarrow \wp(\Sigma^*)$ as follows: For every $\mathcal{O} \subseteq \Sigma^*$, let

$$\mathcal{L}(\mathcal{O}) =_{df} L_{\text{unambiguous}}[\mathcal{O}; N_{i,2}, N_{i,3}, \dots, N_{i,h}].$$

It is easy to see that by Lemma 3.1, \mathcal{L} is (h, t) -ambiguous for polynomial $t(\cdot) =_{df} 5^{h-1} \cdot \prod_{\ell=1}^{h-1} (p \circ)^\ell(\cdot)$. Also, $\mathcal{L}(\mathcal{A} \cup B)$ is defined for every $B \subseteq S_n$ that satisfies $\|B \cap S_{n,f}\| \leq h$ for every $f \in (\Sigma^*)^{\leq k-1}$.

Let $\beta_1, \beta_2, \dots, \beta_{2^k-2}$ be the sequence of nonadaptive queries made by $M_j(0^n)$ to the

oracle $L(N_{i,1}^{\mathcal{L}(\mathcal{A})})$. Consider the following procedure *Diagonalize*.

Procedure Diagonalize($\{\alpha_f \in S_{n,f} \mid f \in (\Sigma^*)^{\leq k-1}\}$)

1. $\mathcal{A}_1 := \mathcal{A}$;
2. For $t := 1$ to $2^k - 1$ do
3. Let $b_{n,1}^{\mathcal{A}_t} b_{n,2}^{\mathcal{A}_t} \dots b_{n,k}^{\mathcal{A}_t}$ be the bit sequence given by (\star) ;
4. If $(b_{n,k}^{\mathcal{A}_t} = 0 \iff M_j^{L(N_{i,1}^{\mathcal{L}(\mathcal{A}_t)})}(0^n)$ accepts) is true then
5. Output \mathcal{A}_t and terminate;
6. Else /* That is, $b_{n,k}^{\mathcal{A}_t} = 0 \iff M_j^{L(N_{i,1}^{\mathcal{L}(\mathcal{A}_t)})}(0^n)$ rejects */
 - (6.1) Let $s := \max\{\ell \in [k] \mid b_{n,\ell}^{\mathcal{A}_t} = 0\}$;
 - (6.2) $\mathcal{A}_{t+1} := \mathcal{A}_t \cup \{\alpha_{b_{n,1}^{\mathcal{A}_t} b_{n,2}^{\mathcal{A}_t} \dots b_{n,s-1}^{\mathcal{A}_t}}\}$; /* That is, flip $b_{n,k}^{\mathcal{A}_t}$. */
 - (6.3) If for every query β_ℓ , it holds that if $N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell)$ rejects then $N_{i,1}^{\mathcal{L}(\mathcal{A}_{t+1})}(\beta_\ell)$ also rejects, then
 Output \mathcal{A}_{t+1} and terminate.
 - (6.4) Else /* there is a query β_ℓ such that $N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell)$ rejects, but $N_{i,1}^{\mathcal{L}(\mathcal{A}_{t+1})}(\beta_\ell)$ accepts. */
 Return to the for loop;

End of Procedure

Claim 7 For each $f \in (\Sigma^*)^{\leq k-1}$, there exists $\widehat{\alpha}_f \in S_{n,f}$ such that for each $t \in [2^k - 1]$ and for each $\ell \in [2^k - 2]$, the following holds in the execution of *Diagonalize*($\{\widehat{\alpha}_f \mid f \in (\Sigma^*)^{\leq k-1}\}$):

if $N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell)$ accepts, then $N_{i,1}^{\mathcal{L}(\mathcal{A}_{t+1})}(\beta_\ell)$ also accepts.

Let us assume that Claim 7 is true. Then there exist strings $\widehat{\alpha}_f \in S_{n,f}$, for each $f \in (\Sigma^*)^{\leq k-1}$, satisfying the property stated in the claim. Set $\mathcal{A} := \text{Diagonalize}(\{\widehat{\alpha}_f \mid f \in (\Sigma^*)^{\leq k-1}\})$ and move to the next stage.

End of Stage

For each $f \in (\Sigma^*)^{\leq k-1}$, let $\widehat{\alpha}_f \in S_{n,f}$ be the strings promised in that claim. Notice that the procedure *Diagonalize*($\{\widehat{\alpha}_f \mid f \in (\Sigma^*)^{\leq k-1}\}$) never adds more than one string in any region $S_{n,f}$. This follows because each region $S_{n,f}$ is associated with exactly one string $\widehat{\alpha}_f \in S_{n,f}$, and only these associated strings are ever considered for inclusion in the oracle. Also note that the effect of Step (6.2) in the procedure is to increment the binary number $b_{n,1}^{\mathcal{A}_t} b_{n,2}^{\mathcal{A}_t} \dots b_{n,k}^{\mathcal{A}_t}$ by 1. That is, we have for each $t \in [2^k - 1]$ considered until the termination of the for loop, $b_{n,1}^{\mathcal{A}_{t+1}} b_{n,2}^{\mathcal{A}_{t+1}} \dots b_{n,k}^{\mathcal{A}_{t+1}} := b_{n,1}^{\mathcal{A}_t} b_{n,2}^{\mathcal{A}_t} \dots b_{n,k}^{\mathcal{A}_t} + 1$. This implies that after the execution of Step (6.2), the bit $b_{n,k}^{\mathcal{A}_t}$ is flipped, i.e., $b_{n,k}^{\mathcal{A}_{t+1}} = \overline{b_{n,k}^{\mathcal{A}_t}}$.

If *Diagonalize*($\{\widehat{\alpha}_f \mid f \in (\Sigma^*)^{\leq k-1}\}$) terminates at Step 5, then clearly $0^n \in L(\mathcal{A}) \iff 0^n \notin L(M_j^{L(N_{i,1}^{\mathcal{L}(\mathcal{A})})})$ and so we successfully finish the stage. Otherwise, *Diagonalize*($\{\widehat{\alpha}_f \mid f \in$

$(\Sigma^*)^{\leq k-1}$) terminates at the execution of Step 6.3 or it terminates because the for loop had finished iterating over the range of values of t .

If $\text{Diagonalize}(\{\widehat{\alpha}_f \mid f \in (\Sigma^*)^{\leq k-1}\})$ terminates at the execution of Step 6.3, then we have the following situation:

- $b_{n,k}^{A_t} = 0 \iff M_j^{L(N_{i,1}^{\mathcal{L}(\mathcal{A}_t)})}(0^n)$ rejects.
- $b_{n,k}^{A_{t+1}} = \overline{b_{n,k}^{A_t}}$.
- For every query β_ℓ , where $\ell \in [2^k - 2]$, if $N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell)$ rejects, then $N_{i,1}^{\mathcal{L}(\mathcal{A}_{t+1})}(\beta_\ell)$ also rejects.
- For every query β_ℓ , where $\ell \in [2^k - 2]$, if $N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell)$ accepts then $N_{i,1}^{\mathcal{L}(\mathcal{A}_{t+1})}(\beta_\ell)$ also accepts, by Claim 7.

It follows that $b_{n,k}^{A_{t+1}} = 1$ if and only if $M_j^{L(N_{i,1}^{\mathcal{L}(\mathcal{A}_{t+1})})}(0^n)$ rejects. Hence, we successfully finish the stage. We next claim that if $\text{Diagonalize}(\{\widehat{\alpha}_f \mid f \in (\Sigma^*)^{\leq k-1}\})$ does not terminate at Step 5, then it must terminate at the execution of Step 6.3.

To this end, for each $t \in [2^k]$, let us define $Q_{\text{acc}}(t)$ to be the set of queries β_ℓ on which $N_{i,1}$ with oracle $\mathcal{L}(\mathcal{A}_t)$ accepts. Formally, for any $t \in [2^k]$, let $Q_{\text{acc}}(t) =_{\text{df}} \{\beta_\ell \mid \ell \in [2^k - 2] \text{ and } N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell) \text{ accepts}\}$. By Claim 7, once a query β_ℓ becomes a member of Q_t , the query β_ℓ remains accepted by $N_{i,1}^{\mathcal{L}(\mathcal{A}_{t'})}$ for any $t \leq t' \in [2^k]$. That is, $Q_{\text{acc}}(t) \subseteq Q_{\text{acc}}(t+1)$ for all $t \in [2^k - 1]$. By the definition of the sets $Q_{\text{acc}}(t)$, it follows that if the condition in Step (6.4) is true at some iteration t of the for loop, then there exist queries $\beta_\ell \in Q_{\text{acc}}(t+1) - Q_{\text{acc}}(t)$; i.e., we have $\|Q_{\text{acc}}(t+1)\| > \|Q_{\text{acc}}(t)\|$ at these iterations t . Thus, there will be an iteration at which the condition in Step (6.4) will not be true. (This follows because the number of iterations, $(2^k - 1)$, of the for loop is greater than the maximum possible size, $2^k - 2$, of $Q_{\text{acc}}(t)$.) Therefore, at that iteration, the condition in Step (6.3) will be true. Hence, $\text{Diagonalize}(\{\widehat{\alpha}_f \mid f \in (\Sigma^*)^{\leq k-1}\})$ will terminate at the execution of Step 6.3. This completes the proof of the theorem. \blacksquare (Theorem 4.13)

Proof of Claim 7. Let f be arbitrary in $(\Sigma^*)^{\leq k-1}$. We will prove that there is a small set $\tilde{Q}(f)$ such that for each $t \in [2^k - 1]$ and for each $\ell \in [2^k - 2]$, the following holds for all $\alpha_f \in S_{n,f} - \tilde{Q}(f)$:

$$\text{if } N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell) \text{ accepts, then } N_{i,1}^{\mathcal{L}(\mathcal{A}_t \cup \alpha_f)}(\beta_\ell) \text{ also accepts.} \quad (4.h)$$

To this end, fix $t \in [2^k - 1]$ and $\ell \in [2^k - 2]$ and assume that $N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell)$ accepts. Let ρ be an arbitrary accepting path ρ in $N_{i,1}^{\mathcal{L}(\mathcal{A}_t)}(\beta_\ell)$. Because \mathcal{L} is (h, t) -ambiguous, for each $z \in \Sigma^*$ there can be at most $t(|z|)$ strings $\alpha_f \in S_{n,f}$ such that

$$z \in \mathcal{L}(\mathcal{A}_t) \iff z \notin \mathcal{L}(\mathcal{A}_t \cup \alpha_f).$$

Since there are at most $p(p(n))$ strings queried on ρ , there is a set $\tilde{Q}(f, t, \ell)$ with $||\tilde{Q}(f, t, \ell)|| \leq p(p(n)) \cdot t(p(p(n)))$ ensuring that $N_{i,1}^{\mathcal{L}(\mathcal{A}_t \cup \alpha_f)}(\beta_\ell)$ accepts for every $\alpha_f \in S_{n,f} - \tilde{Q}(f, t, \ell)$.

It is easy to see that Statement (4.h) is satisfied with $\tilde{Q}(f) = \bigcup_{t,\ell} \tilde{Q}(f, t, \ell)$. Clearly, $||\tilde{Q}(f)|| \leq (2^k - 1) \cdot (2^k - 2) \cdot p(p(n)) \cdot t(p(p(n))) < 2^n$ by our choice of n . Since $||S_{n,f}|| = 2^n$, there exists a string $\widehat{\alpha}_f \in S_{n,f}$ witnessing the correctness of Claim 7. \blacksquare (Claim 7)

Corollary 4.14 *For any integers $k, h \geq 1$, there exists an oracle \mathcal{A} such that*

$$R_{k-T}^p(\text{UP}_{\leq h}^{\mathcal{A}}) \not\subseteq R_{(2^k-2)-tt}^{p,\mathcal{A}}(\text{US}_h^{p,\mathcal{A}}).$$

4.5 Fault-tolerant Access

Ko [Ko87] introduced the notion of *one-sided helping* by a set A in the computation of a set B . A set A is said to provide *one-sided help* to a set B if there is a deterministic oracle Turing machine M computing B and a polynomial $p(\cdot)$ such that (a) on any input $x \in B$, $M^A(x)$ accepts in time $p(|x|)$, and (b) for all inputs y and for all oracles C , $M^C(y)$ accepts (though perhaps $M^C(y)$ may take a longer time than $p(|y|)$) if and only if $y \in B$. Since the machine M , accepting the set B , is capable of answering correctly on faulty oracles, i.e., oracles C different from the oracle A that provides one-sided help to B , the oracle access mechanism is termed fault-tolerant (see [CHV93]). Ko [Ko87] defined $\text{P}_{1\text{-help}}(A)$ to be the class of all sets B that can be one-sided helped by A .

It is known that sets that can be one-sided helped (by any arbitrary helper) are precisely those in NP [Ko87]. Therefore, the notion of one-sided helping provides an avenue for understanding the structure of NP. For instance, given any class $\mathcal{C} \subseteq \text{NP}$, what class of sets in NP can help the computation of sets in \mathcal{C} ? Given any class \mathcal{C}' of helpers, what class $\mathcal{C} \subseteq \text{NP}$ can be helped by sets in \mathcal{C}' ? It is worth studying the relationships between helpers and help-receivers to gain more insights into the notion of one-sided helping.

A restriction of the notion of one-sided helping, called the concept of *helping*, was earlier introduced and studied by Schöning [Sch85]. A set A is said to *help* a set B if B is computed by a deterministic oracle Turing machine M such that on any input $x \in \Sigma^*$, (a) $M^A(x)$ halts in polynomial time, and (b) for all oracles C , $M^C(x)$ accepts (though perhaps $M^C(x)$ may take a longer time than $M^A(x)$ to terminate for $C \neq A$) if and only if $x \in B$.

Definition 4.15 ([Sch85,Ko87]) *1. A deterministic oracle Turing machine M is robust if for all oracles A , M^A halts on each input and $L(M^A) = L(M^\emptyset)$.*

2. A set L is in the class $\text{P}_{1\text{-help}}(A)$ if there exists a robust deterministic oracle Turing machine M and a polynomial $p(\cdot)$ such that $L = L(M^\emptyset)$ and for all $x \in L$, $M^A(x)$ halts in $p(|x|)$ steps. If \mathcal{C} is a complexity class, then $\text{P}_{1\text{-help}}(\mathcal{C}) = \bigcup_{A \in \mathcal{C}} \text{P}_{1\text{-help}}(A)$.

3. A set L is in the class $\text{P}_{\text{help}}(A)$ if there exists a robust deterministic oracle Turing machine M and a polynomial $p(\cdot)$ such that $L = L(M^\emptyset)$ and for all $x \in \Sigma^$, $M^A(x)$ halts in $p(|x|)$ steps. If \mathcal{C} is a complexity class, then $\text{P}_{\text{help}}(\mathcal{C}) = \bigcup_{A \in \mathcal{C}} \text{P}_{\text{help}}(A)$.*

There has been much investigation on the complexity of sets that can be one-sided helped by sets belonging to particular complexity classes. For instance, Ko [Ko87] proved that $\text{NP} = \text{P}_{1\text{-help}}(\text{NP})$, $\text{UP} \subseteq \text{P}_{1\text{-help}}(\text{UP})$, and $\text{P}_{1\text{-help}}(\text{BPP}) \subseteq \text{RP}$. Ko [Ko87] posed the question whether $\text{P}_{1\text{-help}}(\text{UP})$ is exactly the same as UP . Cai, Hemachandra, and Vyskoč [CHV93] proved that relativizable proof techniques cannot resolve this question: There is a relativized world, where $\text{P}_{1\text{-help}}(\text{UP})$ strictly contains UP [CHV93]. Cintioli and Silvestri [CS97] strengthened this result of Cai, Hemachandra, and Vyskoč [CHV93]. They exhibited an oracle A such that $\text{P}_{\text{help}}^A(\text{UP}^A) \not\subseteq \text{Few}^A$ and an oracle B such that $\text{P}_{\text{help}}^B(\text{UP}^B) \not\subseteq R_b^{p,B}(\text{FewP}(n^{\log^{O(1)} n})^B)$, where $\text{FewP}(n^{\log^{O(1)} n})$ is the class of all sets accepted by NPTMs with at most $n^{\log^{O(1)} n}$ accepting paths on inputs of length n . Despite these negative (oracle) results on the provability of containment of $\text{P}_{\text{help}}(\text{UP})$ in classes such as UP , FewP , and Few , Cai, Hemachandra, and Vyskoč [CHV93] were successful in obtaining an exact characterization of $\text{P}_{1\text{-help}}(\text{UP})$. They proved that $\text{P}_{1\text{-help}}(\text{UP})$ is the closure of UP under $\leq_{l_{\text{pos}}}^p$ reductions, where $\leq_{l_{\text{pos}}}^p$ is the polynomial-time *locally positive* reduction introduced by Hemachandra and Jain [HJ91].

We generalize and improve the relativized separation of $\text{P}_{1\text{-help}}(\text{UP})$ from UP by Cai, Hemachandra, and Vyskoč [CHV93] in Corollary 4.17. It remains an open question whether any of the oracle results by Cintioli and Silvestri [CS97], i.e., existence of oracles A and B such that $\text{P}_{\text{help}}^A(\text{UP}^A) \not\subseteq \text{Few}^A$ and $\text{P}_{\text{help}}^B(\text{UP}^B) \not\subseteq R_b^{p,B}(\text{FewP}(n^{\log^{O(1)} n})^B)$, imply our result in Corollary 4.17 for the case of $h = 1$, or vice-versa. It also remains an open question whether the oracle results by Cintioli and Silvestri [CS97] can be generalized so that they hold for $\text{P}_{\text{help}}(\text{UP}_{\leq h})$, for any $h \geq 1$.

Theorem 4.16 *For all $h \geq 1$, there exists an oracle \mathcal{A} such that*

$$R_{\text{dtt}}^p(\text{UP}_{\leq h}^{\mathcal{A}}) \not\subseteq R_{s,b}^{p,\mathcal{A}}(\text{Promise-UP}^{\text{US}_{h-1}^{p,\mathcal{A}}}).$$

Proof The construction of the oracle \mathcal{A} is essentially the same as that in Theorem 4.10 except with few minor changes. For each length n , we will now (1) reserve a segment of n regions $S_{n,f} =_{df} 1^n 01^f 0 \Sigma^{4n}$, where $f \in [n]$, (2) define $S_n =_{df} \bigcup_{f=1}^n S_{n,f}$, and (3) stipulate that $|\mathcal{A} \cap S_{n,f}| \leq h$ for all $f \in [n]$. The rest of the proof is just an imitation of the proof of Theorem 4.10. ■

Corollary 4.17 *For all integer $h \geq 1$, there exists an oracle \mathcal{A} such that*

$$\text{P}_{1\text{-help}}(\text{UP}_{\leq h}^{\mathcal{A}}) \not\subseteq R_{s,b}^{p,\mathcal{A}}(\text{Promise-UP}^{\text{US}_{h-1}^{p,\mathcal{A}}}).$$

Proof This follows from Theorem 4.16, since it can be easily shown that $R_{\text{dtt}}^p(\text{UP}_{\leq h}) \subseteq \text{P}_{1\text{-help}}(\text{UP}_{\leq h})$ in every relativized world.⁵ ■

⁵This same observation was used by Cai, Hemachandra, and Vyskoč [CHV93] (for the case of $h = 1$) in constructing an oracle relative to which $\text{P}_{1\text{-help}}(\text{UP}) \not\subseteq \text{UP}^A$.

5 Robust Unambiguity

So far we looked at several applications of Lemma 3.1 in constructing relativized worlds involving arbitrary levels of the unambiguous polynomial hierarchy. Lemma 3.1, in essence, shows the computational limitations of a $\Sigma_k(A)$ -system under certain weak restrictions. What if we impose a more stringent restriction on a $\Sigma_k(A)$ -system? This question is relevant to our next investigation.

We study the power of robustly unambiguous $\Sigma_k(A)$ -system in Theorem 5.1. (Recall from Definition 2.4 in Section 2, a $\Sigma_k(A)$ -system $[A; N_1, N_2, \dots, N_k]$ is robustly unambiguous if for every oracle B , $[A \oplus B; N_1, N_2, \dots, N_k]$ is unambiguous.) Theorem 5.1 illustrates the following fact: A robustly unambiguous $\Sigma_k(A)$ -system is so weak that given any oracle set B and input x , the hierarchical nondeterministic polynomial-time oracle access to B in $[A \oplus B; N_1, N_2, \dots, N_k](x)$ can be stripped down and turned into a deterministic polynomial-time oracle access (to B) without changing the decision (i.e, acceptance or rejection) of the $\Sigma_k(A \oplus B)$ -system on input x . As a corollary, we obtain a generic oracle collapse of UPH to P assuming $P = NP$. (See, e.g. [BI87, FFKL03] for generic oracles and concepts related to them.)

Theorem 5.1 *For all $A \subseteq \Sigma^*$ and $k \geq 1$, if the $\Sigma_k(A)$ -system $[A; N_1, N_2, \dots, N_k]$ is robustly unambiguous, then for every $B \subseteq \Sigma^*$,*

$$L[A \oplus B; N_1, N_2, \dots, N_k] \in P^{\Sigma_k^{p,A} \oplus B}.$$

Proof The proof is by induction over k . Fix an arbitrary set $A \subseteq \Sigma^*$. The case for $k = 1$ follows by relativization of [HH90, Theorem 2.1]. So suppose that $k > 1$. Notice the following facts: For all sets $B \subseteq \Sigma^*$, we have (a) $L[A \oplus B; N_1, N_2, \dots, N_k] = L[L[A \oplus B; N_2, N_3, \dots, N_k]; N_1]$, (b) $L[L[A \oplus B; N_2, N_3, \dots, N_k]; N_1]$ is unambiguous, and (c) $[A; N_2, N_3, \dots, N_k]$ is robustly unambiguous. Thus by induction hypothesis, we have that for all sets B , $L(A, B) =_{df} L[A \oplus B; N_2, N_3, \dots, N_k] \in P^{\Sigma_{k-1}^{p,A} \oplus B}$. We can now define a nondeterministic polynomial-time Turing machine N'_1 and a set $D \in \Sigma_{k-1}^{p,A}$ such that for all sets $B \subseteq \Sigma^*$,

$$L[L(A, B); N_1] = L[D \oplus B; N'_1], \text{ and } [D; N'_1] \text{ is robustly unambiguous.}$$

It follows by the (strong) induction hypothesis that for every set B , $L[D \oplus B; N'_1] \in P^{NP^D \oplus B} \subseteq P^{NP^{\Sigma_{k-1}^{p,A} \oplus B}}$. Thus the inductive step is proved, since $L[A \oplus B; N_1, N_2, \dots, N_k] = L[D \oplus B; N'_1] \in P^{\Sigma_k^{p,A} \oplus B}$. ■

Corollary 5.2 *If $P = NP$, then relative to a (Cohen) generic G , $P = UPH$.*

The last corollary generalizes a result of Blum and Impagliazzo: If $P = NP$, then relative to a (Cohen) generic G , $P^G = UP^G$ [BI87]. Fortnow and Yamakami [FY96] demonstrated that similar collapses relative to any (Cohen) generic G do not occur at higher levels of the

polynomial hierarchy. They proved that for each $k \geq 2$, there exists a tally set in $\text{UP}^{\Sigma_{k-1}^{p,G}, G} \cap \Pi_k^{p,G}$ but not in $\text{P}^{\Sigma_{k-1}^{p,G}, G}$. Thus Corollary 5.2 contrasts with this generic separation by Fortnow and Yamakami.

6 Conclusion and Open Problems

We presented a counting technique to investigate the structure of relativized hierarchical unambiguous computation. However, two interesting problems have remained open, for whose resolutions the technique presented in this paper could be useful. These problems are:

1. **Simultaneous Immunity and Simplicity in the Relativized Unambiguous Polynomial Hierarchy.**

Complexity class separations can be evaluated in terms of their quality. A separation of a complexity class \mathcal{C}_1 from another class \mathcal{C}_2 by an *immune* set requires the existence of an infinite set L in \mathcal{C}_1 such that no infinite set in \mathcal{C}_2 can be a subset of L ; the set L is called \mathcal{C}_2 -*immune* or *immune* to \mathcal{C}_2 . A separation of a complexity class \mathcal{C}_1 from a class \mathcal{C}_2 by a *simple* set requires the existence of a co-infinite set (i.e., a set whose complement is infinite) L in \mathcal{C}_1 such that L is not in \mathcal{C}_2 and \overline{L} is immune to \mathcal{C}_1 ; the set L is called \mathcal{C}_1 -*simple* or *simple* for \mathcal{C}_1 . Finally, a separation of a complexity class \mathcal{C}_1 from a class \mathcal{C}_2 by a set that is both *simple and immune* requires the existence of a set L in \mathcal{C}_1 such that L is \mathcal{C}_1 -simple and \mathcal{C}_2 -immune.

An oracle separation of a complexity class from another class by a set that is both simple and immune is considered a much more difficult problem than the oracle separations of the same classes by simple sets or by immune sets alone. This point has been discussed in [BT99], which we explain in our own words as follows: “Intuitively, if a set L is \mathcal{C} -immune, then the set L must have low density since no infinite set in \mathcal{C} can be a subset of L . Similarly, if a set L is \mathcal{C} -simple, then the set L must have high density since \overline{L} is \mathcal{C} -immune. Consequently, separation of a complexity class \mathcal{C}_1 from another class \mathcal{C}_2 by a set $L \in \mathcal{C}_1$ that is both \mathcal{C}_1 -simple and \mathcal{C}_2 -immune requires the set L to have conflicting requirements: L must be dense enough so as to be \mathcal{C}_1 -simple and must be thin enough so as to be \mathcal{C}_2 -immune.”

Buhrman and Torenvliet [BT99] showed that relative to an oracle, the first level and the second level of the polynomial hierarchy separate by sets that are both simple and immune. Using Kolmogorov complexity for oracle constructions, they proved that relative to an oracle A , NP has a set that is both NP-simple and $(\text{NP} \cap \text{coNP})$ -immune, and relative to an oracle B , Π_2^p has a set that is both Π_2^p -simple and $(\Sigma_2^p \cap \Pi_2^p)$ -immune. However, it is currently open whether there is an oracle relative to which the third level or any higher level of the polynomial hierarchy separates by a set that is both simple and immune. In fact, it is also open whether there is an oracle relative to which NP has a set that is both NP-simple and coNP-immune.

We expect that the situation for the unambiguous polynomial hierarchy is quite different from the one for the polynomial hierarchy. We hope that it might be possible that an application of our proof technique in conjunction with the Kolmogorov arguments of Buhrman and Torenvliet [BT99] lead to a construction of an oracle relative to which all the levels of the unambiguous polynomial hierarchy separate by sets that are both simple and immune.

2. Random Oracle Separation of the Relativized Unambiguous Polynomial Hierarchy.

There has been an abundance of complexity theoretic results that hold with probability one relative to a random oracle. Some prominent random oracle results are: (1) probability one separation of NP from P with bi-immunity, and of NP from coNP [BG81], (2) probability one separation of NP from P/poly [LS93], (3) probability one separation of coNP from NP with immunity [Ver93], and (4) probability one separation of PSPACE from PH [Cai89,Bab87]. Despite so many random oracle results, the probability one separation of the levels of the polynomial hierarchy relative to a random oracle is still an open problem. (See [HRZ95] for an extensive discussion on this problem.) Currently, only the circuit complexity-theoretic approach is known for separating the higher levels (levels beyond three) of the polynomial hierarchy, but the circuit approach has so far not been successful in resolving this longstanding open problem.

We believe that the case of the unambiguous polynomial hierarchy is easier. In Theorem 4.1, we have used our counting technique to show that for all $k \geq 1$, there is an oracle A such that $UP_{\leq k+1}^A$ is not contained in $U\Sigma_k^{p,A}$. Thus, unlike the case of the polynomial hierarchy for which only the circuit approach is known for the relativized separation of all its levels, the levels of the relativized unambiguous polynomial hierarchy are separable by counting arguments alone, and thus by completely avoiding the machinery of circuit complexity. This raises our hope that a probability one separation of the levels of UPH might be easier to achieve than its counterpart, i.e., a probability one separation of the levels of PH relative to a random oracle.

Acknowledgment We thank Lane Hemaspaandra and Jörg Rothe for their constant encouragement and support.

References

- [ACRW04] S. Aida, M. Crăsmaru, K. Regan, and O. Watanabe. Games with uniqueness properties. *Theory of Computing Systems*, 37(1):29–47, 2004.
- [AK02] V. Arvind and P. Kurur. Graph isomorphism is in SPP. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, pages 743–750, Los Alamitos, November 2002. IEEE Computer Society.

- [Bab87] L. Babai. A random oracle separates PSPACE from the Polynomial Hierarchy. *Information Processing Letters*, 26(1):51–53, 1987.
- [BC93] D. Bovet and P. Crescenzi. *Introduction to the Theory of Complexity*. Prentice Hall, 1993.
- [Bei89] R. Beigel. On the relativized power of additional accepting paths. In *Proceedings of the 4th Structure in Complexity Theory Conference*, pages 216–224. IEEE Computer Society Press, June 1989.
- [Bei91] R. Beigel. Bounded queries to SAT and the boolean hierarchy. *Theoretical Computer Science*, 84(2):199–223, 1991.
- [Bei93] R. Beigel. The polynomial method in circuit complexity. In *Proceedings of the 8th Structure in Complexity Theory Conference*, pages 82–95, San Diego, CA, USA, May 1993. IEEE Computer Society Press.
- [BG81] C. Bennett and J. Gill. Relative to a random oracle A , $P^A \neq NP^A \neq coNP^A$ with probability 1. *SIAM Journal on Computing*, 10:96–113, 1981.
- [BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the $P=?NP$ question. *SIAM Journal on Computing*, 4(4):431–442, 1975.
- [BI87] M. Blum and R. Impagliazzo. Generic oracles and oracle classes. In *Proceedings of the 28th IEEE Symposium on Foundations of Computer Science*, pages 118–126, October 1987.
- [BS79] T. Baker and A. Selman. A second step toward the polynomial hierarchy. *Theoretical Computer Science*, 8:177–187, 1979.
- [BST93] H. Buhrman, E. Spaan, and L. Torenvliet. Bounded reductions. In K. Ambos-Spies, S. Homer, and U. Schöning, editors, *Complexity Theory*, pages 83–99. Cambridge University Press, 1993.
- [BT99] H. Buhrman and L. Torenvliet. Complicated complementations. In *Proceedings of the 14th Annual IEEE Conference on Computational Complexity*, pages 227–236, Los Alamitos, May 4–6 1999. IEEE Computer Society.
- [BU98] C. Berg and S. Ulfberg. A lower bound for perceptrons and an oracle separation of the PP^{PH} hierarchy. *Journal of Computer and System Sciences*, 56(3):263–271, 1998.
- [Cai89] J. Cai. With probability one, a random oracle separates PSPACE from the polynomial-time hierarchy. *Journal of Computer and System Sciences*, 38(1):68–85, 1989.
- [CGRS04] M. Crăsmaru, C. Glaßer, K. Regan, and S. Sengupta. A protocol for serializing unique strategies. In *Proceedings of the 29th International Symposium on Mathematical Foundations of Computer Science*. Springer-Verlag *Lecture Notes in Computer Science* #3153, August 2004.

- [CHV92] J. Cai, L. Hemachandra, and J. Vyskoč. Promise problems and access to unambiguous computation. In *Proceedings of the 17th Symposium on Mathematical Foundations of Computer Science*, pages 162–171. Springer-Verlag *Lecture Notes in Computer Science #629*, August 1992.
- [CHV93] J. Cai, L. Hemachandra, and J. Vyskoč. Promises and fault-tolerant database access. In K. Ambos-Spies, S. Homer, and U. Schöning, editors, *Complexity Theory*, pages 101–146. Cambridge University Press, 1993.
- [CS97] Cintioli and Silvestri. Helping by unambiguous computation and probabilistic computation. *Theory of Computing Systems*, 30:165–180, 1997.
- [ESY84] S. Even, A. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, 1984.
- [FFK94] S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. *Journal of Computer and System Sciences*, 48(1):116–148, 1994.
- [FFKL03] S. Fenner, L. Fortnow, S. Kurtz, and L. Li. An oracle builder’s toolkit. *Information and Computation*, 182(2):95–136, 2003.
- [FSS84] M. Furst, J. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.
- [FY96] L. Fortnow and T. Yamakami. Generic separations. *Journal of Computer and System Sciences*, 52(1):191–197, February 1996.
- [Gol05] O. Goldreich. On promise problems. Technical report TR05–018, Electronic Colloquium on Computational Complexity (ECCC), 2005.
- [GS88] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988.
- [GT05] C. Glaßer and S. Travers. Machines that can output empty words. Technical report TR05–147, Electronic Colloquium on Computational Complexity (ECCC), 2005.
- [Gup92] S. Gupta. On the closure of certain function classes under integer division by polynomially bounded functions. *Information Processing Letters*, 44:205–210, 1992.
- [Hås87] J. Håstad. *Computational Limitations of Small-Depth Circuits*. MIT Press, 1987.
- [HH90] J. Hartmanis and L. Hemachandra. Robust machines accept easy sets. *Theoretical Computer Science*, 74(2):217–225, 1990.
- [HJ91] L. Hemachandra and S. Jain. On the limitations of locally robust positive reductions. *International Journal of Foundations of Computer Science*, 2(3):237–255, 1991.

- [HO02] L. Hemaspaandra and M. Ogihara. *The Complexity Theory Companion*. Springer, 2002.
- [HR97] L. Hemaspaandra and J. Rothe. Unambiguous computation: Boolean hierarchies and sparse Turing-complete sets. *SIAM Journal on Computing*, 26(3):634–653, 1997.
- [HRZ95] L. Hemaspaandra, A. Ramachandran, and M. Zimand. Worlds to die for. *SIGACT News*, 26(4):5–15, 1995.
- [JY85] D. Joseph and P. Young. Some remarks on witness functions for non-polynomial and non-complete sets in NP. *Theoretical Computer Science*, 39:225–237, 1985.
- [Ko85] K. Ko. On some natural complete operators. *Theoretical Computer Science*, 37(1):1–30, 1985.
- [Ko87] K. Ko. On helping by robust oracle machines. *Theoretical Computer Science*, 52:15–36, 1987.
- [Ko89] K. Ko. Relativized polynomial time hierarchies having exactly k levels. *SIAM Journal on Computing*, 18(2):392–408, 1989.
- [LR94] K.-J. Lange and P. Rossmanith. Unambiguous polynomial hierarchies and exponential size. In *Proceedings of the 9th Structure in Complexity Theory Conference*, pages 106–115. IEEE Computer Society Press, June/July 1994.
- [LS93] J. Lutz and W. Schmidt. Circuit size relative to pseudorandom oracles. *Theoretical Computer Science*, 107:95–120, 1993.
- [NR98] R. Niedermeier and P. Rossmanith. Unambiguous computations and locally definable acceptance types. *Theoretical Computer Science*, 194(1–2):137–161, 1998.
- [OH93] M. Ogiwara and L. Hemachandra. A complexity theory for feasible closure properties. *Journal of Computer and System Sciences*, 46(3):295–325, 1993.
- [Pap94] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [Reg97] K. Regan. Polynomials and combinatorial definitions of languages. In L. Hemaspaandra and A. Selman, editors, *Complexity Theory Retrospective II*, pages 261–293. Springer-Verlag, 1997.
- [Sch85] U. Schöning. Robust algorithms: A different approach to oracles. *Theoretical Computer Science*, 40:57–66, 1985.
- [Sip83] M. Sipser. Borel sets and circuit complexity. In *Proceedings of the 15th ACM Symposium on Theory of Computing*, pages 61–69. ACM Press, 1983.
- [SL96] M. Sheu and T. Long. UP and the low and high hierarchies: A relativized separation. *Mathematical Systems Theory*, 29(5):423–449, 1996.

- [ST] H. Spakowski and R. Tripathi. On the power of unambiguity in alternating machines. *Theory of Computing Systems*. To appear.
- [Ver93] N. Vereshchagin. Relationships between NP-sets, co-NP-sets, and P-sets relative to random oracles. In *Proceedings of the 8th Structure in Complexity Theory Conference*, pages 132–138. IEEE Computer Society Press, May 1993.
- [Wag90] K. Wagner. Bounded query classes. *SIAM Journal on Computing*, 19(5):833–846, 1990.
- [Yao85] A. Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, pages 1–10, 1985.